Comparison of Two Bandpass
Delta-Sigma A/D Converter Architectures

by

Sevgi Ertan

Submitted to the
Department of Electrical Engineering and Computer Science

in Partial Fulfillment of the Requirements
for the Degree of

Master of Engineering in Electrical Engineering and Computer Science

at the Massachusetts Institute of Technology

January 2000

Signature of Author _____

Sevgi Ertan
Department of Electrical Engineering and Computer Science
January 14, 2000

Approved By _____

Paul Ward
Technical Supervisor, Draper Laboratory

Certified By _____

Professor Charles G. Sodini
Thesis Supervisor, Massachusetts Institute of Technology

Accepted By _____

Arthur C. Smith
Chairman, Department Committee on Graduate Thesis

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	13.Jun.00	THESIS

4. TITLE AND SUBTITLE
COMPARISON OF TWO BANDPASS DELTA-SIGMA A/D CONVERTER ARCHITECTURES

5. FUNDING NUMBERS

6. AUTHOR(S)
2D LT ERTAN SEVGI

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
THE DEPARTMENT OF THE AIR FORCE
AFIT/CIA, BLDG 125
2950 P STREET
WPAFB OH 45433

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
FY00-210

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Unlimited distribution
In Accordance With AFI 35-205/AFIT Sup 1

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

14. SUBJECT TERMS

15. NUMBER OF PAGES
78

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

Comparison of Two Bandpass
Delta-Sigma A/D Converter Architectures

by

Sevgi Ertan

Submitted to the
Department of Electrical Engineering and Computer Science

January 14, 2000

In Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Electrical Engineering and Computer Science

ABSTRACT

This study compares the performance and robustness of single loop and cascaded bandpass delta-sigma analog-to-digital (A/D) converter architectures. A 4^{th} order single loop modulator and a 4^{th} order 1-1 cascaded architecture are designed to provide a basis for comparison of SNR performance and robustness to capacitor mismatch and finite operational amplifier gain variations. Several different approaches to the design of error cancellation circuitry in the cascaded architecture are presented.

MATLAB simulations of the two architectures is used to show that the single loop modulator yields better SNR performance than the cascaded modulator. Both architectures perform comparably under capacitor mismatch variations. The effects of operational amplifier gain on both architectures are insignificant for gains greater than about 8,000.

Thesis Supervisor: Charles G. Sodini
Title: Professor of Electrical Engineering and Computer Science

ACKNOWLEDGEMENTS

I would like to first and foremost, thank my mother, Nazmiye, for all her love, affection, and help that she has selflessly given to me throughout my whole life. My friend through both thick and thin, my teacher of both technical matters and life, without her support nothing I have ever accomplished in my life would have been possible.

I would also like to thank my Thesis Advisor, Professor Charlie Sodini, my Academic Advisor, Bernard Lesieutre, and my Technical Supervisors at Draper Laboratory, Paul Ward and Ed Balboni, for their technical support and guidance through out my work on this project. I am also grateful to the staff at Draper who made working there enjoyable, and in particular, to Ralph Haley and Ochida Martinez, who never failed to listen and offer supporting words.

Finally, I would like to thank my friends in general who made studying at MIT bearable and at times very enjoyable. In particular, I would like to thank Keith Santarelli, Louis Nervegna, and Phil Juang for their support, encouragement, and the many laughs during stressful times, both during my thesis work as well as during classes.

Knowledge should mean a full grasp of knowledge;
Knowledge means to know yourself, heart and soul.
If you have failed to understand yourself,
Then all of your reading has missed its call

What is the purpose of reading those books?
So that Man can know the All-Powerful.
If you have read, but failed to understand,
Then your efforts are just a barren toil

Don't boast of reading, mastering science
Or of all your prayers and obeisance.
If you know not who God is
All of your learning is of no use at all.

Yunus Emre says to you, pharisee,
Make the holy pilgrimage if need be
A thousand times - but if you ask me,
The visit to a heart is best of all.

- Yunus Emre (13th Century Turkish Poet)

This thesis was prepared at The Charles Stark Draper Laboratory, Inc, under Internal Company Sponsored Research No. C312.

Sevgi Ertan

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Analog-to-Digital Conversion

There are many methods for performing analog-to-digital conversion. Conventional methods, such as flash A/D or successive approximation converters, sample their analog inputs at the Nyquist rate (i.e. twice the signal bandwidth). While such circuits have the advantage of simpler architectures, they also have several features that make them undesirable in large scale integrated circuits. These include requiring high precision analog components and stringent anti-aliasing requirements.

Through the use of oversampling, however, complexity in the analog domain can be traded off for fast and more complex digital signal processing [1, 2, 3]. Because VLSI technology is better suited for fast digital circuits than precise analog circuits, high performance can be achieved using a low cost CMOS process. Furthermore, by oversampling, the quantization noise is spread out over a larger frequency band, so that the total amount of noise in the frequency band of interest is diminished.

A baseline topology for an N-th order oversampling converter is a single-loop structure like the one depicted in Figure 1.1. In general, the quantizer can have many levels. Oversampling converters that use feedback are known as delta-sigma modulators [2].

The transfer function for a typical quantizer is shown in Figure 1.2. From a large scale perspective, the quantizer function seems to have a linear gain G, that clips at the

maximum output level, Δ/2. When the input signal to the quantizer results in a clipped output, the quantizer is said to be overloaded. At a finer level, however, the quantizer is

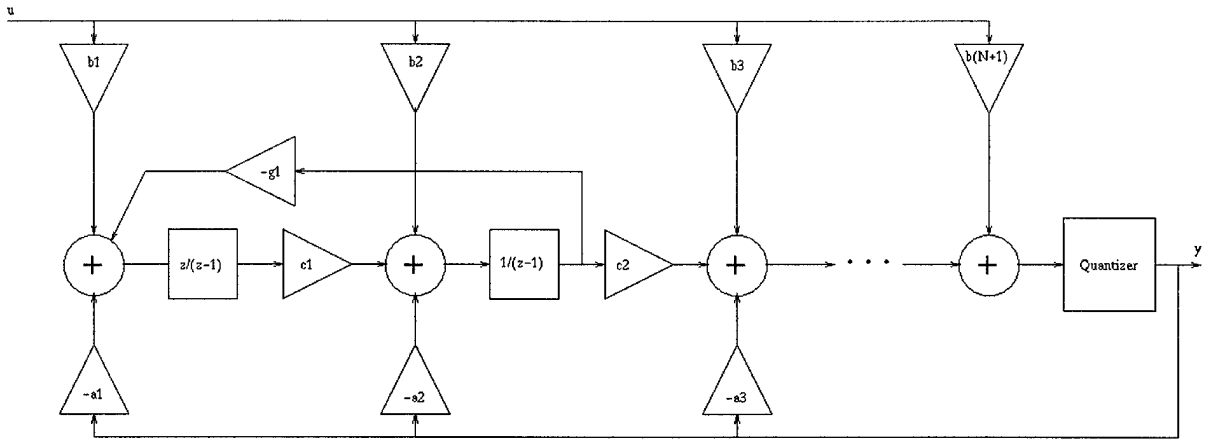

Figure 1.1: N-th Order Single Loop Delta-Sigma Modulator

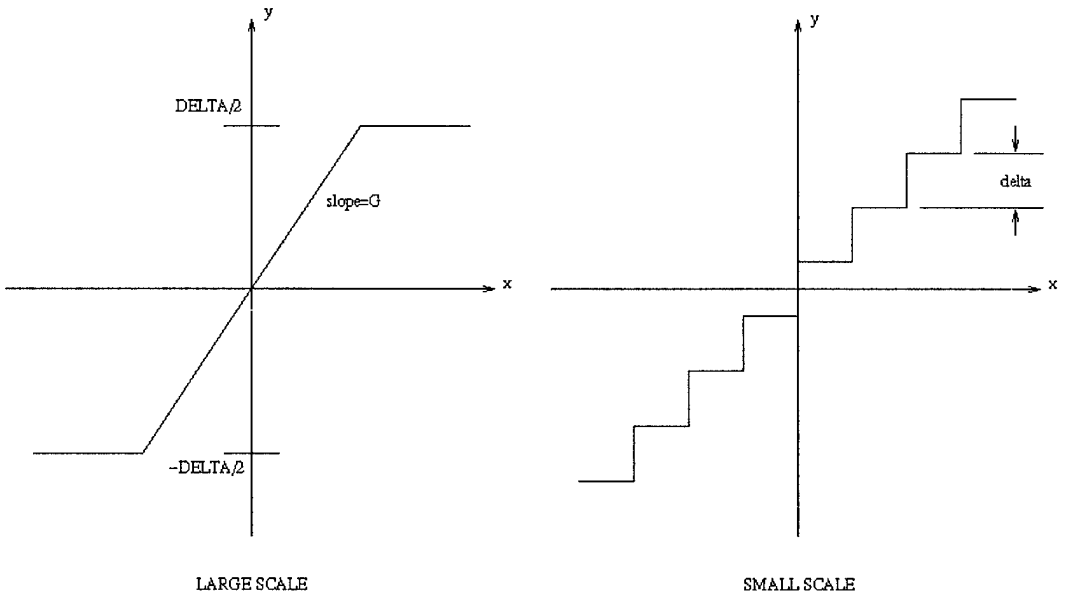

Figure 1.2: Quantizer Transfer Function

truly a step function. For B bits of quantizer resolution, the difference between output levels, δ, is given by $\delta=\Delta/(2^B-1)$.

Thus, the quantizer output can be represented by the input scaled by the gain plus an error signal:

$$y[n] = Gx[n] + e(x[n]) \tag{1.1}$$

The nonlinear nature of the quantizer makes the A/D system difficult to analyze. The simplest model of the quantizer is the unity gain approximation: an approximate analysis based on a statistical estimate of the quantizer error which states that the quantizer error can be modelled as a unity gain with an additive white noise source if the following conditions are met [4, 5, 6]:

- The quantizer is not overloaded;
- The quantizer has a large number of quantization levels;
- The quantizer level separation, δ, is small relative to the signal level; and
- The joint probability density of any two quantizer input samples is smooth.

The unity gain approximation, however, is not the only possible quantizer model. More intricate models using describing functions and the root locus method have been developed to take into account the variation of the quantizer gain under normal operation [4, 7]. Nevertheless, for a first order analysis and basic understanding of the operation of oversampling converters, the unity gain approximation is sufficient. Thus, under this approximation, the overall output of the converter can be expressed as

$$Y(z) = H_x(z)U(z) + H_e(z)E(z) \tag{1.2}$$

where U(z) and Y(z) are the z-transforms of the input and output signals respectively, and E(z) is the power spectral density of the quantizer error under the white noise approximation. $H_x(z)$ is referred to as the signal transfer function (STF) and $H_e(z)$ is referred to as the noise transfer function (NTF). Oversampling converters are designed such that in the frequency band of interest, the STF magnitude is approximately 1 and the NTF magnitude is much less that 1 in the band of interest. Out-of-band noise is amplified, but this noise is digitally filtered later. Modulators whose passband include only low frequencies are referred to as lowpass modulators, whereas modulators whose passband is between two nonzero frequencies, f_1 and f_2 are referred to as bandpass modulators.

By increasing the order of the noise shaping, the NTF attenuates more quantization noise in-band and the output of the modulator becomes a more accurate representation of the input. The main difficulty with such higher order, single loop structures, however, is stability. The stability requirements for a single loop modulator are [8]:

- The poles of the linearized transfer function must be inside the unit circle;
- The integrator amplifiers should not saturate;
- All integrators must have an initial state of zero; and
- The quantizer should not overload.

For a 1-bit quantizer, however, the gain of the quantizer is a function of the input, and is therefore not determinate. Thus, the system is only conditionally stable and stability may depend on the amplitude of the input signal or on precise circuit matching. Through the

addition of control circuitry or the proper design of modulator coeffecients, however, higher order structures can be stabilized, and are widely used in many designs [9].

An alternative solution to the stability problem is to instead use multi-stage noise shaping (MASH) and cascade stable first or second order single-loop structures, as shown in Figure 1.3. The output of one stage becomes the input to the next stage after passing through what is known as an "error mixing network." Lambda is the error mixing

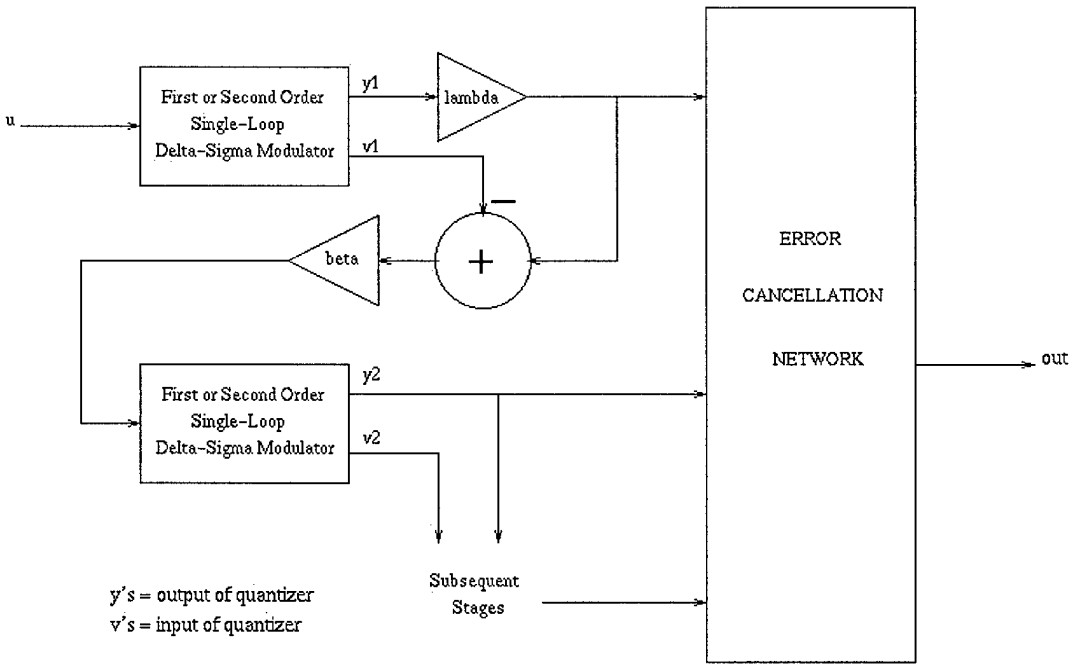

Figure 1.3: MASH Delta-Sigma Modulator

coefficient, and beta is the error gain coefficient. Notice that if lambda=1, the input to the next stage is simply a scaled version of the quantization error of the previous stage. To eliminate the noise due to the additional quantizers, such cascaded structures also require

error cancellation circuitry, or filters which are designed to cancel out the quantization noise of subsequent stages.

The chief problem with cascaded architectures, however, is that they can be more sensitive to capacitor mismatch errors and other non-idealities. Any error in the coefficients of the error cancellation circuitry result in the increase of the quantization noise that is seen at the output.

1.2 Background Research

The motivation for this research stems from the need at Draper Laboratory for an A/D converter with good resolution and high dynamic range to process the narrowband output signal of a precision sensor. There are three principle ways of approaching the conversion of a narrowband signal. One way is to mix the signal down to DC, and then perform lowpass delta-sigma modulation (Figure 1.4). The chief problem with this approach, however, is that the output is susceptable to drift in the DC value of the input to the A/D converter as well as to 1/f noise.

Another approach is to apply a low pass delta-sigma conversion prior to demodulation (Figure 1.5). However, this is not the optimal approach for processing the narrowband signal because the noise in the band of interest may only experience moderate suppression. Such designs have provided reasonable performance, but have not yet met the desired noise level specifications.

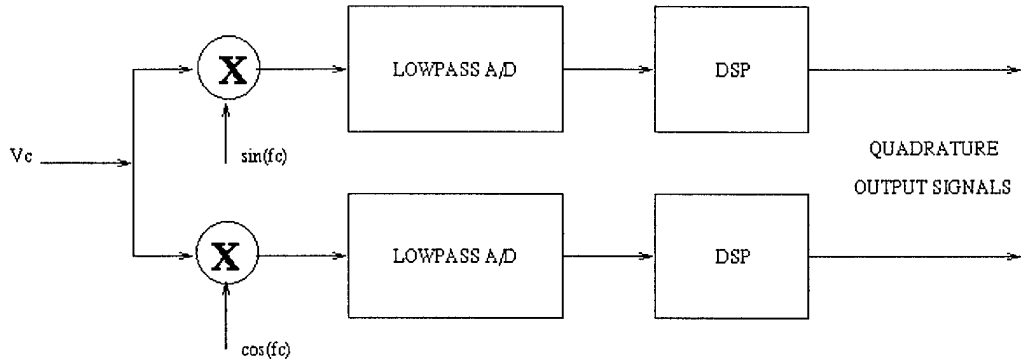

Figure 1.4: Narrowband Signal Processing:
Mix signal down to DC, then use lowpass modulator

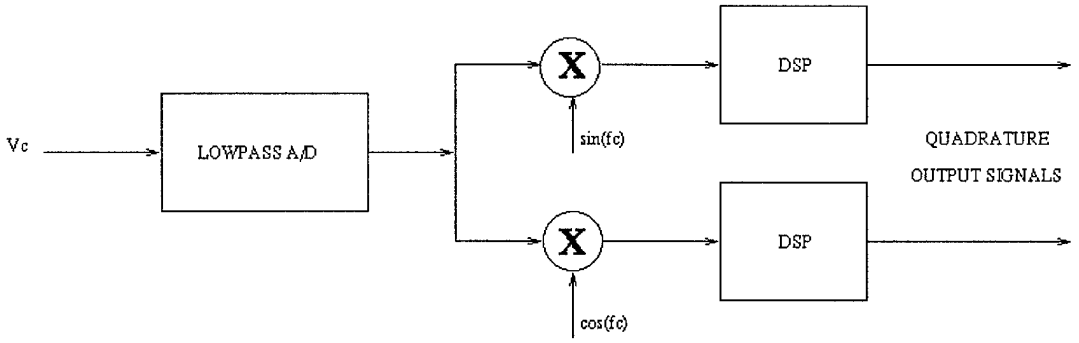

Figure 1.5: Narrowband Signal Processing:
Directly convert signal with lowpass modulator

A third way of processing the signal is to use a bandpass delta-sigma converter (Figure 1.6), which takes advantage of the narrowband nature of the signal to improve performance and power dissipation.

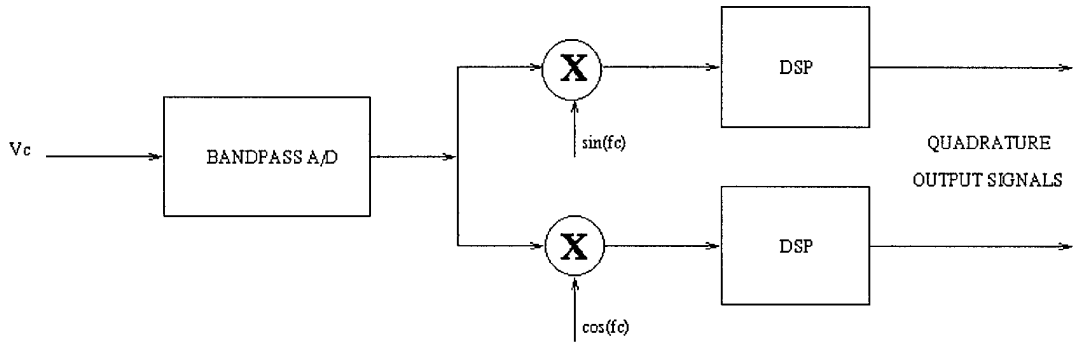

Figure 1.6: Narrowband Signal Processing:
Directly convert signal with bandpass modulator

Previous delta-sigma designs at Draper Laboratory have all utilized the second conversion method: direct application of a lowpass delta-sigma converter. Indeed, much of the literature focuses on lowpass delta-sigma conversion using a single-loop topology. However, cascaded architectures have gained increasing attention. A 1994 work by David Ribner [10] introduced bandpass cascaded architectures at a theoretical level, and comparative studies on a MATLAB level of the influence of non-idealities and quantization noise have been done by other researchers [11, 12, 13, 14]. The performance of some verified modulator designs and their performance is summarized in Table 1. The modulator requirements of Draper Laboratory are also included. As a basis

for comparing modulators, a performance factor based on Signal-to-Noise Ratio (SNR), bandwidth, and power consumption was devised:

$$\text{SNR-Hz/W} = \text{SNR} + 10\log(\text{Bandwidth}) - 10\log(\text{Power Dissipation}) \quad (1.3)$$

	Ref[15]	**Ref[16]**	**Ref[17]**	**Ref[18]**	**Ref[19]**	**Ref[20]**	**Ref[21]**	**Draper (Specs)**
Samp. Freq. (Hz)	1.852M	42.8M	8M	160M	10M	80M	4M	1.28M
Carrier Freq. (Hz)	445k	10.7M	2M	5M	3.75M	20M	-	20k-25k
Bandwidth	8k	200k	30k	2.5M	30k	200k	25k	200
SNR (dB)	63	57	56	84	77	72	99	114
Power Diss. (mW)	480	60	0.8	1000	130	49	2.5	< 10
Area (mm^2)	24.5	-	1	10	4.32	2.73	1.5	< 5
Technology	-	-	2u CMOS	0.8u BiCMOS	0.8u BiCMOS	0.6u CMOS	0.8u CMOS	0.6u CMOS
SNR-Hz/W (dB)	105	122	132	148	131	138	169	157
Architecture	Bandpass Single-Loop	Bandpass Single-Loop	Bandpass Single-Loop	Bandpass Single-Loop	Bandpass Single-Loop	Bandpass Single-Loop	Lowpass Cascade	-

Table 1: Summary of Past Modulator Designs

1.3 Thesis Objective

The objective of this thesis is to further explore bandpass delta-sigma conversion, and in particular MASH architectures, in light of the system requirements of Draper Laboratory. Thus, two architectures are considered in this work:

1. A fourth order single-loop bandpass delta-sigma modulator; and
2. A fourth-order 1-1 cascaded bandpass delta-sigma modulator.

Both are analyzed with regards to their design issues, and the effects of non-idealities, in particular capacitor mismatch and finite operational amplifier gain.

CHAPTER 2

A 4-TH ORDER SINGLE LOOP Δ-Σ MODULATOR

2.1 Topology

A block diagram for the 4th order single loop architecture is shown in Figure 2.1. From this block diagram, the NTF for this architecture can be found to be

$$NTF = \frac{\left[z^2 + (g_1c_1 - 2)z + 1\right]\left[z^2 + (g_2c_3 - 2)z + 1\right]}{z^4 + k_1z^3 + k_2z^2 + k_3z + k_4} \tag{2.1}$$

where

$$k_1 = g_1c_1 + g_2c_3 - 4 + c_4[a_4 + a_3c_3]$$

$$k_2 = 2 + (g_1c_1 - 2)(g_2c_3 - 2) + c_4[c_2c_3(a_2 + a_1c_1) + (g_1c_1 - 2)(a_4 + a_3c_3) - a_4]$$

$$k_3 = g_1c_1 + g_2c_3 - 4 + c_4[a_4 + a_3c_3 - c_2c_3a_2 - a_4(g_1c_1 - 2)]$$

$$k_4 = 1 - c_4a_4$$

The primary consideration in selecting the modulator coefficients is choosing the NTF such that the in-band SNR is maximized. As can be observed from the NTF expression, for bandpass modulators, NTF zeros come in complex conjugate pairs. As a result, only even-order bandpass systems are realizable. One possible choice is to place all the NTF zeros in the center of the band. In this case, high SNR is achieved near the center frequency, to the detriment of system performance at the fringes of the band. For

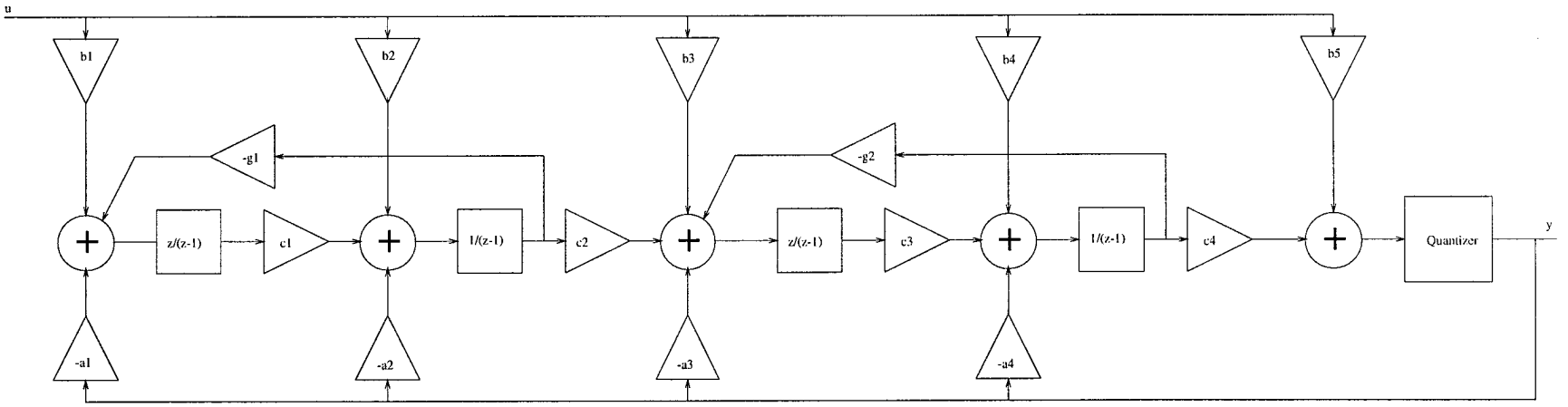

Figure 2.1: 4th Order Single Loop Architecture

wideband systems, this is not an optimal placement. If the zero pairs are separated, and placed away from the center, then the performance at freqencies at the fringes of the band is improved at the expense of performance near the center of the band. At a certain point, for each system, there is an optimal placement of poles away from the center. For small bandwidth systems, however, coincident zeros at the center frequency yield optimal performance because most the input signals are concentrated at the center frequency anyway.

The narrowband system required by Draper Laboratory has a bandwidth of 200Hz. Thus, all NTF zeros were placed at the center frequency of 20kHz. To facilitate the design of the remaining modulator coefficients, a Delta-Sigma Toolbox developed by Richard Schreier for MATLAB was used. A description of the Toolbox commands used is included in Appendix A. In this way, a set of coefficients that realizes the desired NTF with unity STF was computed. However, to ensure correct operation of the modulator, the integrators must not be saturated. Thus, the modulator coefficients must then be scaled according to the input amplitude range to prevent clipping. This was again accomplished using MATLAB to yield the following modulator coefficients and NTF:

a_1=0.3964	a_2=0.3754	a_3=0.4079	a_4=0.3890	
b_1=0.3964	b_2=0.3754	b_3=0.4079	b_4=0.3890	b_5=1
c_1=0.0863	c_2=0.2361	c_3=0.4242	c_4=1.4277	
g_1=0.1116	g_2=0.0227			

$$NTF = \frac{(z^2 - 1.99z + 1)^2}{(z^2 - 1.491z + 0.5652)(z^2 - 1.687z + 0.7865)} \tag{2.2}$$

2.2 Predicted SNR

A primary performance measure of A/D converters is the signal-to-noise ratio (SNR). The SNR is defined as the ratio of the output signal power to the noise power. Consider the SNR calculation for an L-th order real bandpass delta-sigma modulator with an input signal band of $\pm f_B$ about a carrier f_c and a sampling rate of f_s.

$$SNR \equiv 10\log\left(\frac{\sigma_x^2}{\sigma_{ey}^2}\right) \tag{2.3}$$

where σ_x^2 is the input signal power and σ_{ey}^2 is the in-band noise power at the output. Under the unity gain approximation, the quantizer error, e[n], is modelled as white noise. Further assuming that the modulator output is filtered by an ideal bandpass filter (to remove all out-of-band noise), the in-band noise power can be expressed as

$$\sigma_{ey}^2 = \int_{band} \frac{\sigma_e^2}{f_s} |H_e(f)|^2 df \tag{2.4}$$

where $\sigma_e^2 = \frac{V^2}{3}$ and V is the amplitude of the input sinusoidal signal.

Using these two equations, the SNR ratio for any system, given the NTF, H_e, can be found. A detailed derivation of the SNR for an L^{th}-order bandpass modulator can be found in [22] to yield the following result:

$$SNR = 10\log\frac{3a^2(L+1)(2R)^{L+1}}{4k^2\pi^2}dB \tag{2.5}$$

where L is the order of the modulator, and R is the oversampling ratio ($f_s/2f_b$),

$k = \frac{1}{L(L-1)...1}\frac{d^L\left|H(e^{j\omega})\right|}{d\omega^L}\Big|_{\omega=\omega_o}$ and ω_o and ω_b are the center frequency and bandwidth (in radians) respectively.

To calculate the expected SNR of the 4th order modulator specified in the previous section, a MATLAB function called *Predict4.m* was written. Using *Predict4.m*, the expected SNR at the output is computed to be 185.8 dB. The code for *Predict4.m* is included in Appendix B.

Note that these SNR predictions consider the noise due to quantization noise only. In an actual implementation of a delta-sigma modulator, other circuit noise sources also factor into the overall SNR. However, for the purposes of comparing the single loop and cascaded architectures, only the SNR due to quantization noise is considered.

2.3 Capacitor Mismatch and Finite Op Amp Gain

Typically the integrators of the modulator are implemented using switched capacitor circuitry. An implementation of z/(z-1) is shown in Figure 2.2 and an implementation of 1/(z-1) is shown in Figure 2.3. Note that these circuits are identical except their sampling clocks are reversed.

To assess the effects of finite op amp gain and capacitor mismatch, the transfer function for the overall circuit must be computed in terms of the op amp gain (A) and the

capacitor values. The assumptions about the op amp used in this calculaton are discussed in the next section.

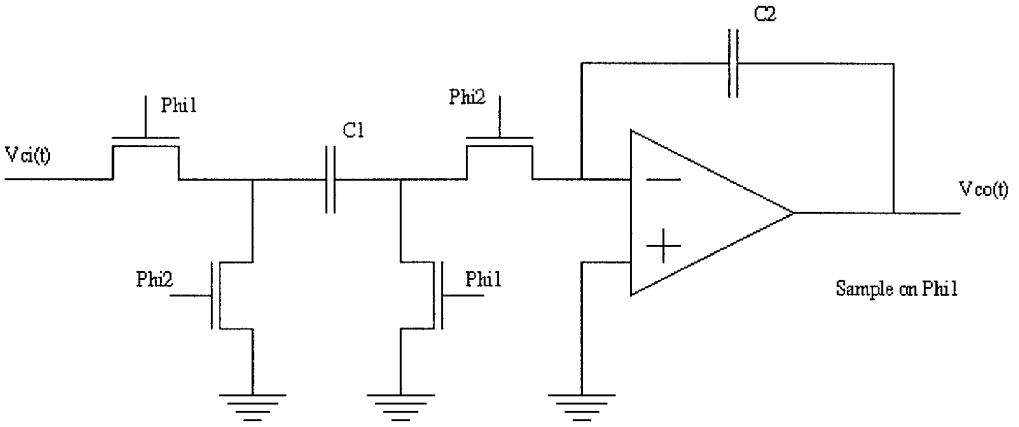

Figure 2.2: Switched Capacitor Implementation of z/(z-1)

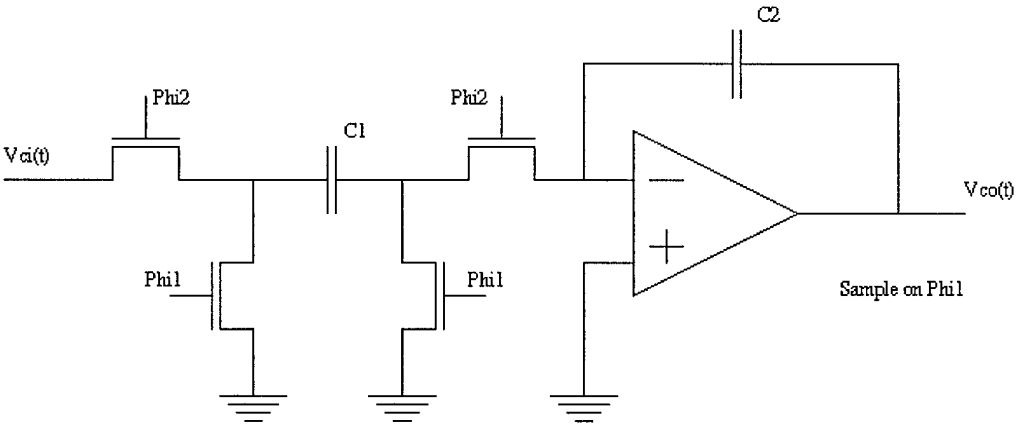

Figure 2.3: Switched Capacitor Implementation of 1/(z-1)

2.3.1 Op Amp Modeling

An ideal op amp is modelled as having a constant, but very large, open loop gain A, such that the output of the op amp, V_o, is related to its input voltages, V_- and V_+ by

$$V_o = A(V_+ - V_-) \tag{2.6}$$

This equation, however, ignores the frequency dependency of the op amp gain. A typical op amp has a very large gain at DC, and at the -3dB frequency (which defines the bandwidth of the op amp) begins to roll off at -20 dB/decade, until finally dropping to unity at the crossover frequency. Once past the crossover frequency, the gain falls further and additional high frequency poles cause the gain to roll off faster. The phase of the op amp is also not a constant. At DC, the phase is zero, whereas by the crossover frequency the phase is at least -90°. Beyond crossover, higher frequency poles cause the phase to shift even more negative.

A key performance factor of an op amp is its gain-bandwidth (G-B) product. Because the G-B product is a constant at a particular temperature, an op amp with a high DC gain has a smaller bandwidth than an op amp with a low DC gain. Depending on the design of the op amp then, the system may either be operating in the constant gain region below the 3 dB frequency, or in the 20 dB rolloff region between the 3 dB and crossover frequency. Thus, when assessing the effects of finite op amp gain, the gain that must be considered is not the DC gain A_o but the gain at the operating frequency $A(f_c)$. In simulations, $A(f_c)$ will be considered to vary from as low as 100 to as high as 5,000 and for simplicity, the phase is assumed to be zero.

2.3.2 Transfer Function Considering Non-Idealities

With this op amp model in mind, the transfer function for the implementation of z/(z-1), the circuit of Figure 2.2, in terms of capacitor ratios and op amp gain can be found to be

$$\frac{V_{co}(z)}{V_{ci}(z)} = \frac{\frac{AC_1}{C_1+(1+A)C_2}z^{-1}}{1-\frac{(1+A)C_2}{C_1+(1+A)C_2}z^{-1}} \rightarrow \frac{\frac{C_1}{C_2}z^{-1}}{1-z^{-1}} \quad \text{when A goes to infinity} \qquad (2.7)$$

Similarly, the transfer function for the implementation of 1/(z-1), the circuit of Figure 2.3, is found to be

$$\frac{V_{co}(z)}{V_{ci}(z)} = \frac{\frac{-AC_1}{C_1+(1+A)C_2}}{1-\frac{(1+A)C_2}{C_1+(1+A)C_2}z^{-1}} \rightarrow \frac{-\frac{C_1}{C_2}}{1-z^{-1}} \quad \text{when A goes to infinity.} \qquad (2.8)$$

By adjusting the state-space equation coefficients according to the transfer functions derived above, the effects of non-idealities are introduced in the MATLAB simulations. The results of such simulations are presented in Chapter 4.

CHAPTER 3

A 4-TH ORDER 1-1 MASH Δ-Σ MODULATOR

3.1 Topology

A block diagram for the 4th order 1-1 MASH architecture is shown in Figure 3.1. The structure consists of two cascaded 2nd order single loop structures. However, the arhitecture is known as a 1-1 MASH because each stage contains one resonator in the forward path. In general, the two stages can have different modulator coefficients. However, in this system, because both stages are being sampled at the same rate, identical second order stages were used.

Using an approach identical to that used in the design of the 4th order single loop modulator, the coefficients for the 2nd order sections comprising each stage of the MASH architecture were designed to be:

$a_1 = a_3 = 0.4846$ $a_2 = a_4 = 0.1261$

$b_1 = b_4 = 0.4846$ $b_2 = b_5 = 0.1261$ $b_3 = b_6 = 1.0000$

$c_1 = c_3 = 0.0992$ $c_2 = c_4 = 4.4286$

$g_1 = g_2 = 0.0971$

which gives a noise transfer function of

$$NTF = \frac{z^2 - 1.99z + 1}{z^2 - 1.219z + 0.4413} \tag{3.1}$$

To prevent saturation of the second stage, λ and β were chosen to be 1 and 0.8, respectively.

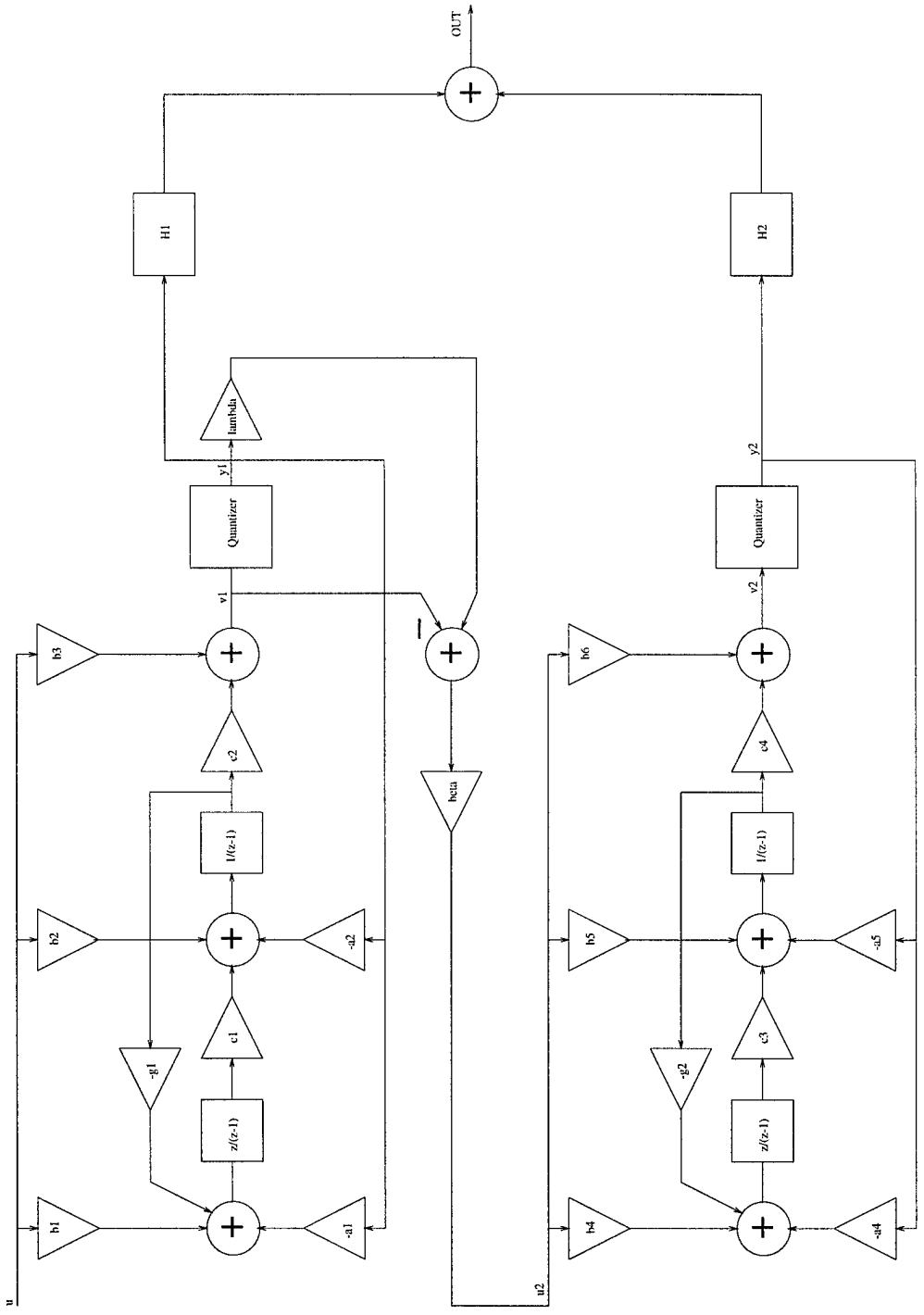

Figure 3.1: 4th Order MASH Architecture

3.2 Error Cancellation Filter Design

3.2.1 Exact Noise Cancellation

The error cancellation filters, H_1 and H_2, are designed such that the quantization noise from the quantizer of the first stage is cancelled. Again, using the unity gain approximation, the following relations can be found from the block diagram:

$$Y_1 = \frac{STFn}{STFd}U + \frac{NTFn}{NTFd}E_1 \tag{3.2}$$

$$Y_2 = \frac{STFn}{STFd}U_2 + \frac{NTFn}{NTFd}E_2 \tag{3.3}$$

$$U_2 = \beta(\lambda Y_1 - V_1) = \beta[(\lambda - 1)Y_1 - E_1] \tag{3.4}$$

$$OUT = H_1 Y_1 + H_2 Y_2 \tag{3.5}$$

where U, U_2, Y_1, Y_2, V_1, OUT, β and λ are as indicated in Figure 4.1; STFn and STFd are the numerator and denominator polynomials of the STF respectively; NTFn and NTFd are the numerator and denominator polynomials of the NTF respectively, and E_1 and E_2 are the power spectral densities of the quantization error of the first and second stage quantizers respectively.

From these equations, OUT can be calculated in terms of U_1, E_1 and E_2:

$$OUT = \left(H_1 + \beta(\lambda - 1)\frac{STFn}{STFd}H_2\right)\frac{STFn}{STFd}U \tag{3.6}$$

$$+\left[\left(H_2 + \beta(\lambda - 1)\frac{STFn}{STFd}H_2\right)\frac{NTFn}{NTFd} + \beta\frac{STFn}{STFd}H_2\right]E_1 + \frac{NTFn}{NTFd}H_2 E_2$$

However, because the STF is designed to be 1 this equation reduces to

$$OUT = (H_1 + \beta(\lambda - 1)H_2)U + \left[(H_1 + \beta(\lambda - 1)H_2)\frac{NTFn}{NTFd} + \beta H_2 \right] E_1 + \frac{NTFn}{NTFd} H_2 E_2 \quad (3.7)$$

To cancel the quantization noise from the first stage quantizer, the coefficient of E_1 is set to zero to yield the following requirement on H_1 and H_2:

$$(H_1 + \beta(\lambda - 1)H_2)NTFn + \beta(NTFd)H_2 = 0 \quad (3.8)$$

Substituting the expressions for the NTF, this equation can be solved to find the H_2 required for exact noise cancellation in terms of H_1:

$$H_2 = \left(\frac{-1}{\beta} \right) \frac{1 + k_3 z^{-1} + z^{-2}}{\lambda + (k_1 + (\lambda - 1)k_3)z^{-1} + (k_2 + \lambda - 1)z^{-2}} H_1 \quad (3.9)$$

where $NTFn=1+k_3z^{-1}+z^{-2}$ and $NTFd=1+k_1z^{-1}+k_2z^{-2}$. From this equation, we can see that one obvious choice for H_1 and H_2 would be:

$$H_1 = -\beta\left(\lambda + (k_1 + (\lambda - 1)k_3)z^{-1} + (k_2 + \lambda - 1)z^{-2}\right) \quad \text{and} \quad H_2 = 1 + k_3 z^{-1} + z^{-2} \quad (3.10)$$

To calculate the SNR predicted by the theory, a MATLAB function called *Model.m* was written. This function takes as an input a sinusoid at the center frequency

and processes it according to the block diagram and the unity gain approximation: the input signal is filtered by the STF and NTF of each stage to compute Y_1 and Y_2, which were then filtered by H_1 and H_2 and summed to compute the overall modulator output, OUT. Using *Model.m*, the expected SNR at the output is computed to be 193 dB. A plot of the output of the first stage and the modulator is shown in Figure 3.2.

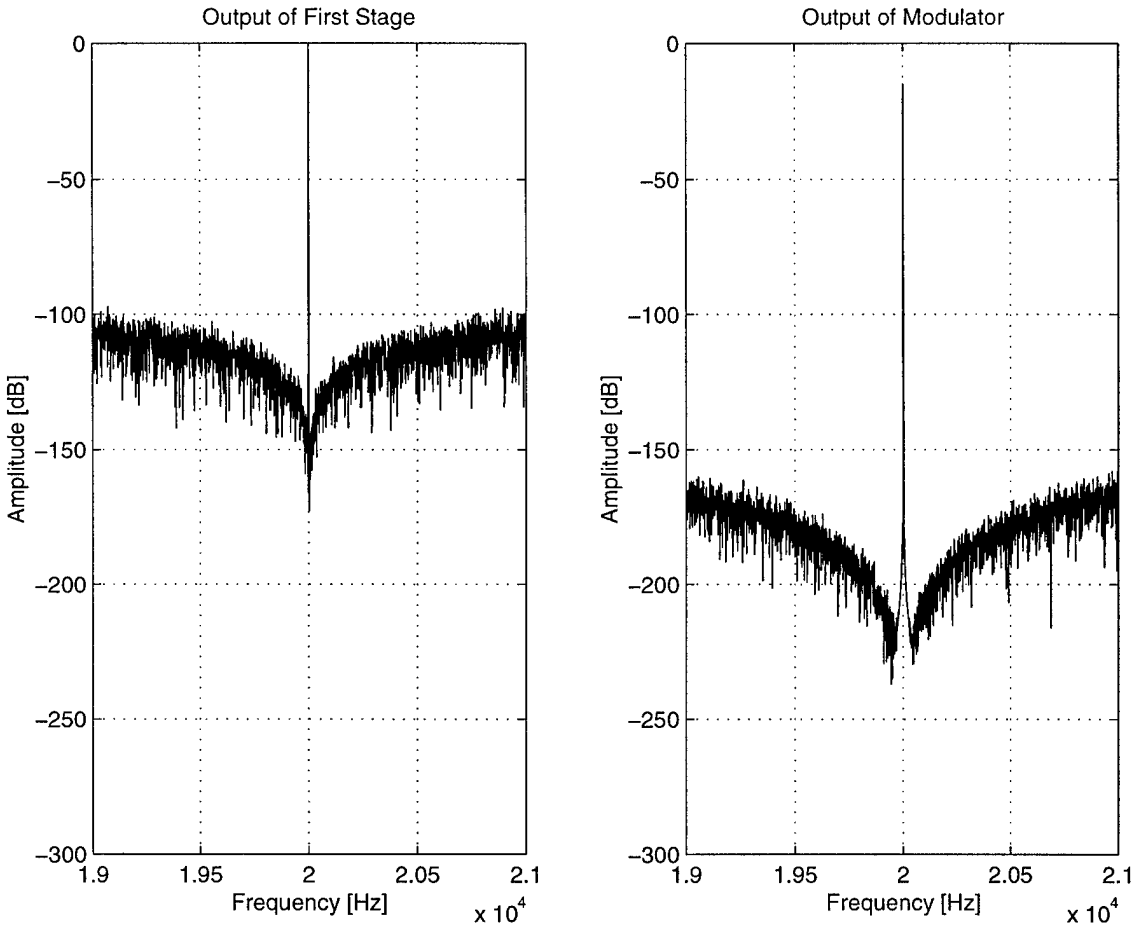

Figure 3.2: Power Spectrum of 1st Stage and Modulator Output Signals for Exact Error Cancellation Filters

The chief problem with this selection of filters, however, is apparent from the spectrum of the modulator output: the noise is suppressed to levels at about -210 dB, but signal amplitude as also been attenuated by about 20 dB! The reason for this can be seen from the equation for OUT: for $\lambda=1$, the coefficient of U is H_1. Because H_1 is itself a filter which attenuates in-band frequencies, the input signal is also attenuated.

To avoid this problem, H_1 should be set to a unit delay. In this case, an exact cancellation of the quantization error requires that H_2 be an Infinite Impulse Response (IIR) filter. However, using an IIR filter in the error cancellation network results in a continuous, as opposed to quantized output, because the output can then take on values in between the quantization levels. To preserve the quantized nature of the output, H_2 must be selected to be a Finite Impulse Response (FIR) filter.

However, there is no FIR filter for H_2 that can exactly satisfy equation (3.8) when H_1 is selected to be a unit delay!

Thus, the design of H_2 reduces to a minimization problem: given that the error cannot be totally cancelled, what filter minimizes this error? There are many possible filters that could be designed in answer to this question. This study looks at two design approaches: one based on modelling the system in the frequency domain, and one based on modelling the system in the time domain.

3.2.2 Frequency Response Approximation

One way to approach this problem is to model the frequency response of $P \equiv \frac{1}{\lambda + (k_1 + (\lambda - 1)k_3)z^{-1} + (k_2 + \lambda - 1)z^{-2}}$ as an FIR filter. A plot of the frequency response is shown in Figure 3.3.

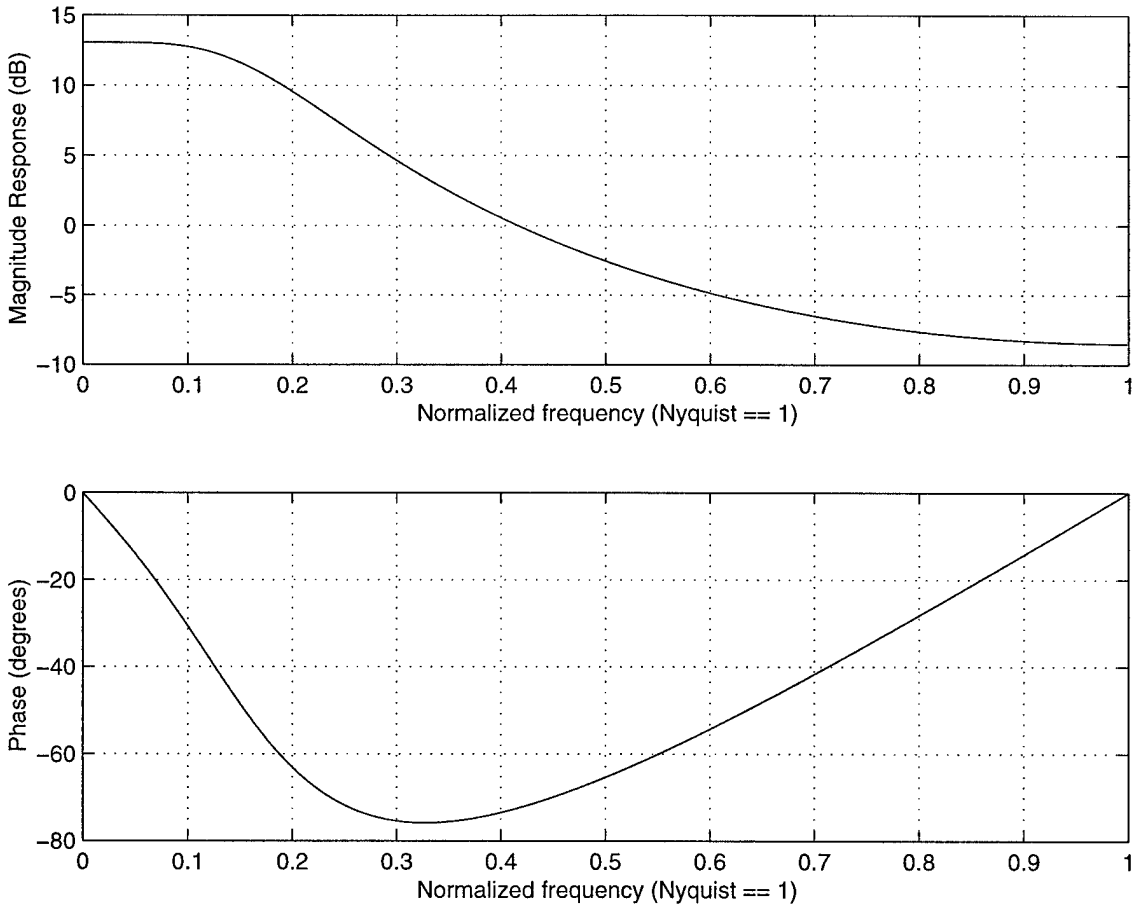

Figure 3.3: Freqency Response of P(z)

From this plot it can be observed that the magnitude of P(z) is essentially constant in the frequency band of interest. Approximating P(z) as a constant gain (henceforth referred to as the Gain Model) H_2 can be found to be

$$H_2 = -\frac{K}{\beta}\left(1 + k_6 z^{-1} + z^{-2}\right)H_1 \qquad \text{where K is a constant.} \qquad (3.11)$$

From the magnitude plot of the frequency response, K was found to be 4.5, giving error cancellations filters of

$$H_1 = z^{-1} \qquad \text{and} \qquad H_2 = -5.625z^{-1} + 11.196z^{-2} - 5.625z^{-3} \qquad (3.12)$$

With this selection of error cancellation filters, *Model.m* computes the SNR to be 145 dB.

A better approximation that adds a minimal amount of complexity involves also taking into account the phase information of the frequency response (Gain-Phase Model). At low frequencies, the phase drops linearly. Modeling P(z) as Kz^{-1} leads to error cancellation filters of

$$H_1 = z^{-1} \qquad \text{and} \qquad H_2 = -5.625z^{-2} + 11.196z^{-3} - 5.625z^{-4} \qquad (3.13)$$

With these error cancellation filters, *Model.m* computes the SNR to be 149 dB over a 200 Hz bandwidth - a 5 dB improvement over the Gain Model for P(z). A plot of the ideal performance with this selection of error cancellation filters is shown in Figure 3.4.

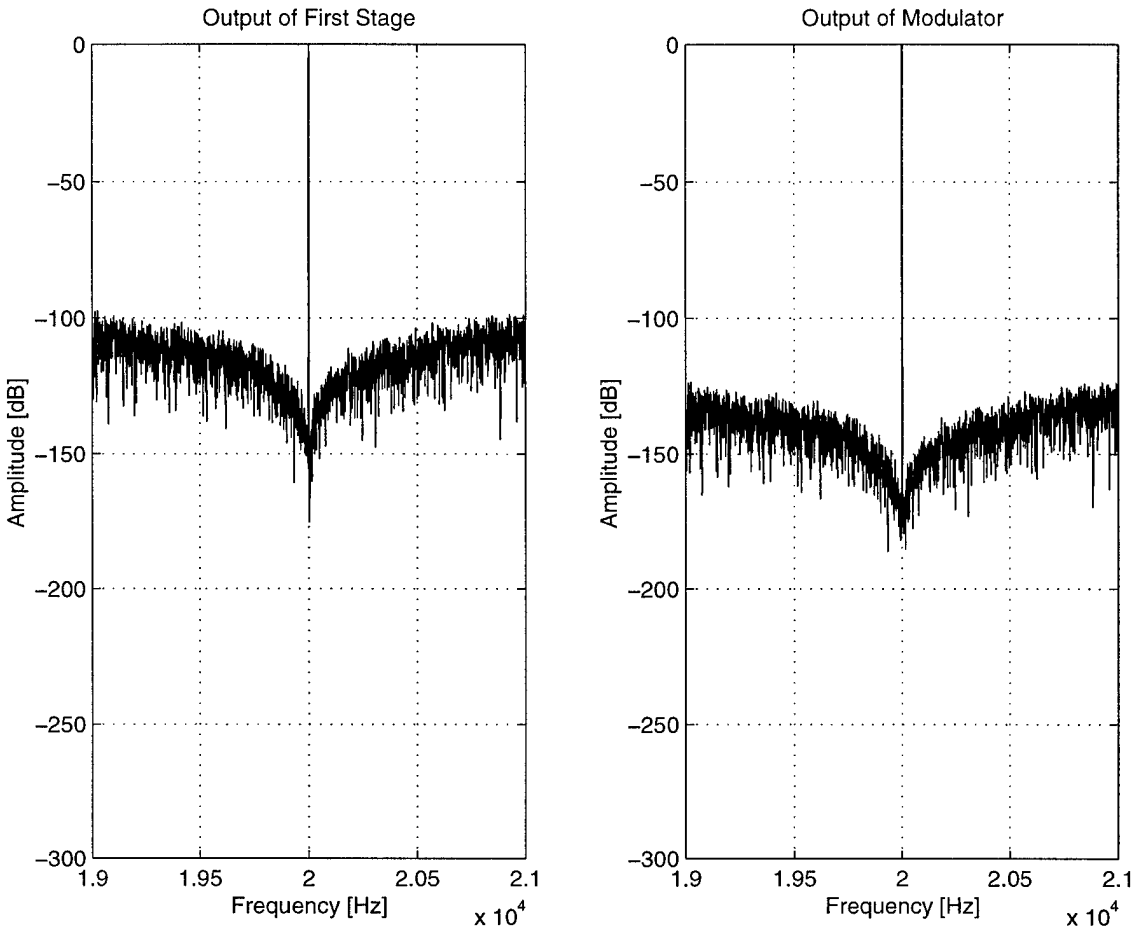

Figure 3.4: Power Spectrum of 1st Stage and Modulator Output Signals Using Gain-Phase Model

3.2.3 Impulse Response Approximation

Another design approach involves using the impulse response of P(z) to approximate P(z) as an FIR filter. A plot of the impulse response of P(z) is shown in Figure 3.5. From this plot, it can be observed that altough P(z) is an IIR filter, after a delay of 17 units, the response has died down so that P(z) can be well represented by

$$P(z) \approx \sum_{n=0}^{17} k_n z^{-n} \tag{3.14}$$

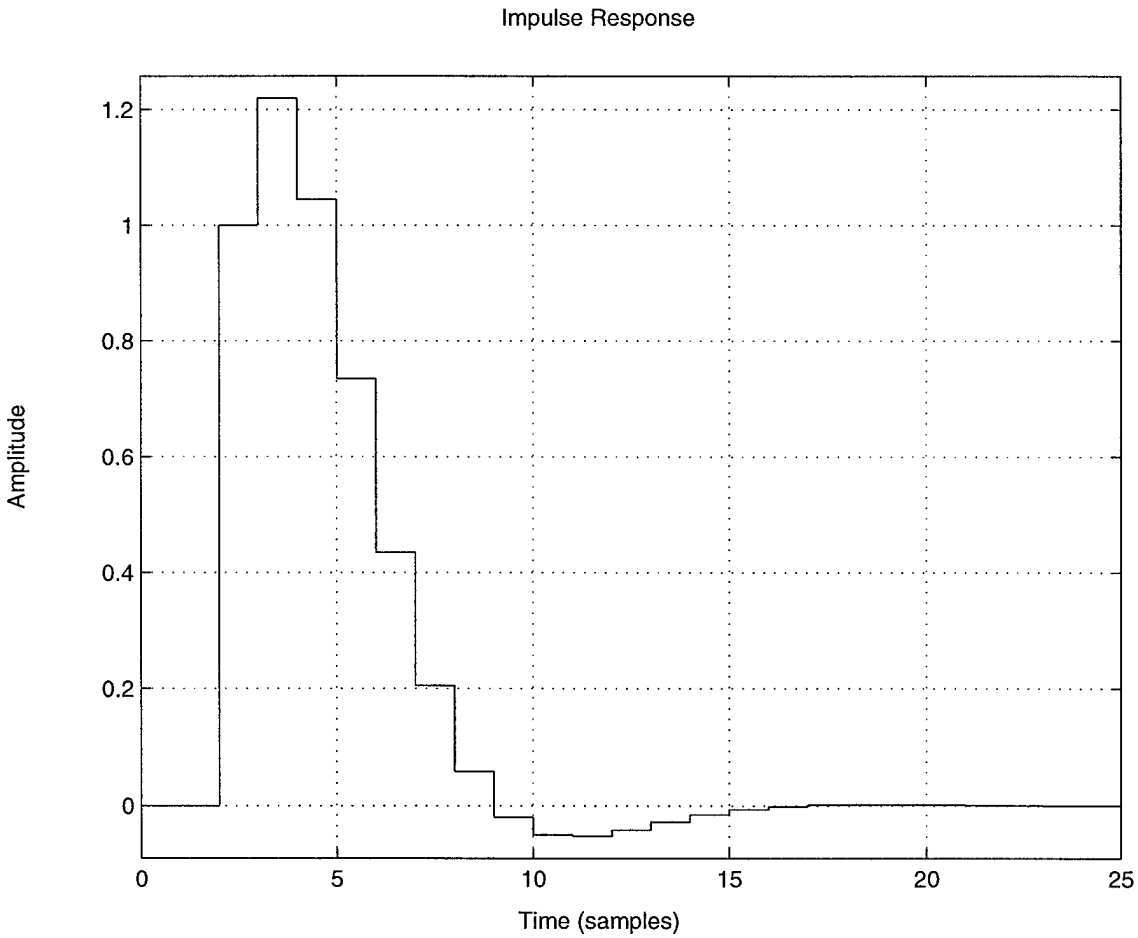

Figure 3.5: Impulse Response of P(z)

The coefficients k_n can be directly read off from the impulse response plot. If P(z) is approximated with all 18 coefficients, the result will be very good - near exact - error cancellation. Choosing fewer coefficients may result in a simpler system, but is also a poorer approximation of the impulse response.

To get an idea for the tradeoff between complexity and performance, the SNR was computed using *Model.m* for n=1 to n=17. The results are shown in Table 2.

	n	SNR [dB]
Tier 1:		
	1	135
	2	137
	3	142
	4	150
Tier 2:		
	5	163
	6	163
	7	159
	8	160
	9	163
	10	167
Tier 3:		
	11	181
	12	181
	13	196
	14	186
	15	185
	16	187
	17	189

Table 2: SNR Calculated for Varying Filter Lengths

From the table we can observe roughly three levels of performance (labelled as Tiers 1, 2 and 3). The filters in Tier 1 have the lowest order. While being simpler in that filters with less coefficients generally require less hardware to implement them, these

filters do not perform well in comparison with the Gain-Phase Model based filter presented in the previous section.

The performance of the Tier 2 filters is better than that of the Gain-Phase Model filter, and has an SNR 20 dB higher than the filters in Tier 1. By moving up to the Tier 3 filters, those that use the most coefficients of the impulse response, results very close to those that would have been attained with the exact error cancellation filter, had not the signal level also been attenuated, are achieved.

3.2.4 Error Cancellation Filter Selection

The design of error cancellation filters involves a tradeoff of performance with efficiency and robustness. Higher order filters have better SNR performance, but also generally require more hardware to implement, thereby increasing the power consumption of the overall system. Additionally, systems using error cancellation filters with more coefficients are generally also more sensitive to capacitor mismatch variations than those that use filters with fewer coefficients, because more parameters are being affected by the non-ideality.

For the purposes of this study, the primary MASH architecture considered utilizes the Gain-Phase Model based error cancellation filters:

$$H_1 = z^{-1} \qquad \text{and} \qquad H_2 = -5.625z^{-2} + 11.196z^{-3} - 5.625z^{-4} \tag{3.15}$$

However, because the Tier 3 filters are expected to yield SNR performance comparable to that of the single loop architecture, the error cancellation filter designed for the model with n=17 is also considered:

$$H_1 = z^{-1}$$

and

$$\begin{aligned} H_2 = & -1.25z^{-1} + 0.9642z^{-2} + 0.4773z^{-3} + 0.1561z^{-4} - 0.0205z^{-5} - 0.0939z^{-6} \\ & - 0.1054z^{-7} - 0.087z^{-8} - 0.0596z^{-9} - 0.0342z^{-10} - 0.0154z^{-11} - 0.0036z^{-12} + 0.0024z^{-13} \\ & + 0.0045z^{-14} + 0.0044z^{-15} + 0.0034z^{-16} + 0.0022z^{-17} + 0.0039z^{-18} - 0.0029z^{-19} \end{aligned} \tag{3.16}$$

CHAPTER 4

SIMULATION RESULTS

4.1 Single Loop Architecture

4.1.1 Simulated Performance

MATLAB was used to simulate the modulator and test performance. All MATLAB code used for simulating the single loop architecture is included in Appendix B. The bulk of the code consists of using iteration to compute the output sequence given a sinusoidal input sequence. The quantizer is implemented as a comparator.

The SNR of the modulator is calculated by using a test input sinusoid of 20kHz (the center frequency). The power spectrum of the output sequence is calculated. The SNR is given by the difference in decibels of the signal amplitude and the in-band noise. Thus, the SNR is found to be 186.4 dB in a 200 Hz bandwidth.

This simulation result compares well with the 185.8 dB SNR predicted earlier.

4.1.2 Effects of Capacitor Mismatch

To assess the effects of capacitor mismatch and finite op amp gain, the equations derived in the previous chaper were coded into MATLAB. Assuming an op amp gain of 1,000,000, capacitor mismatch variations were added to the nominal values. For variations of 10^{-4} (+/- 1% 1σ), 10^{-6} (+/- 0.1% 1σ), and 10^{-8} (+/- 0.01% 1σ) the system was tested over 100 runs, for which the SNR was recorded each time. The graph in Figure 4.1

summarizes the results of the data. The individual histograms from which this plot was generated are included in Appendix D.

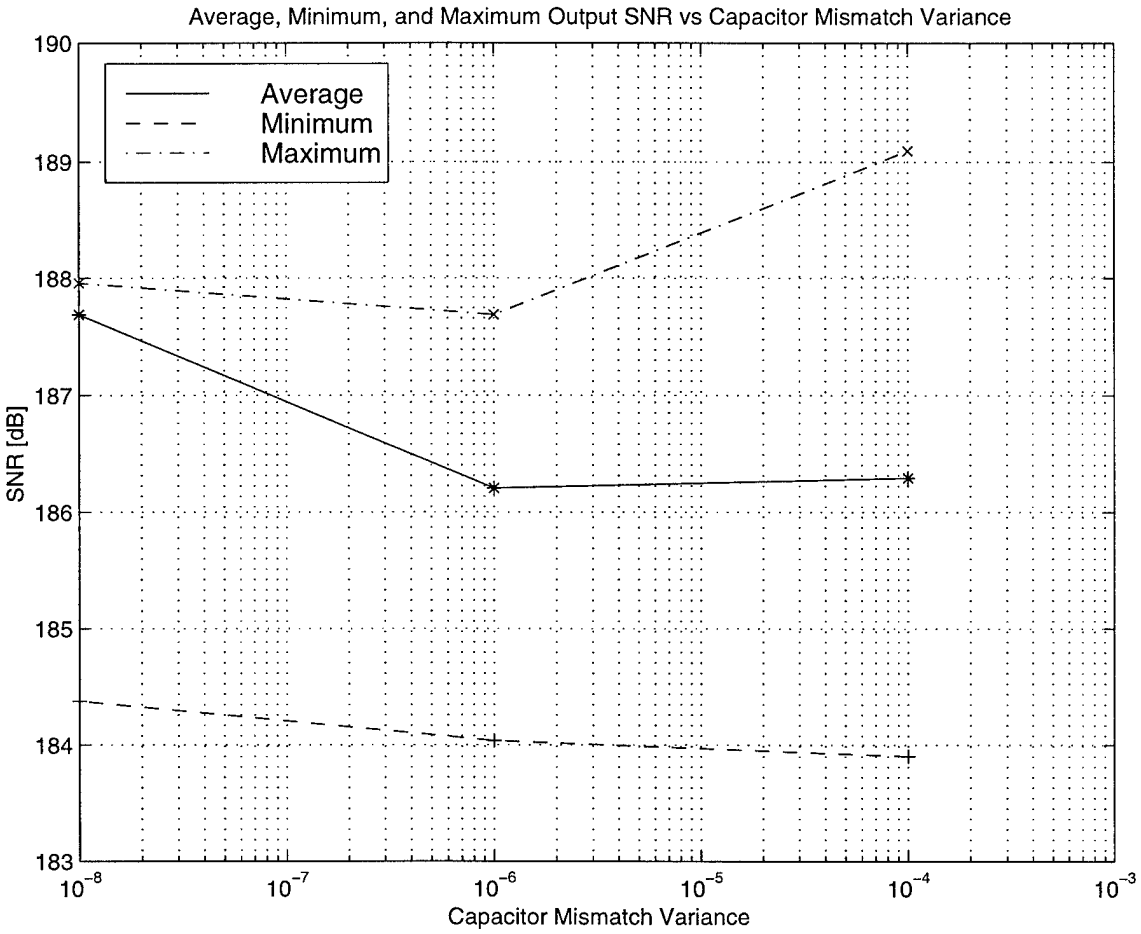

Figure 4.1: Capacitor Mismatch Simulation Results for Single Loop Architecture

4.1.3 Effects of Finite Op Amp Gain

The effects of finite op amp gain was determined by calculating the SNR of the modulator for the following gains: 100, 300, 600, 1000, 3000, 5000, and 10000. The results of these simulations is plotted in Figure 4.2.

This plot demonstrates that in order to achieve near the ideal performance of the modulator, the op amp gain at the operating frequency must be kept above about 3,000.

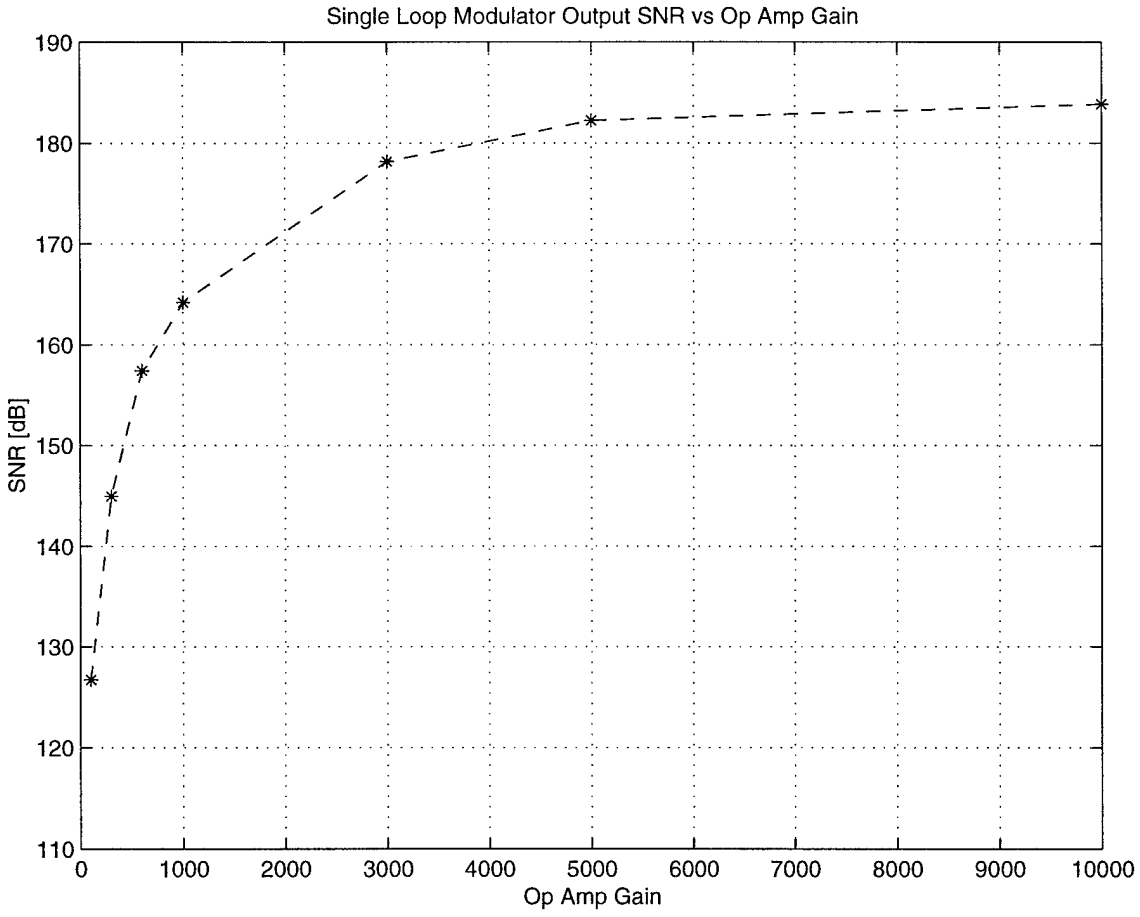

Figure 4.2: Plot of Single Loop Modulator SNR With Varying Op Amp Gain

4.2 MASH Architecture

With the same methodology as applied for the single-loop modulator, MATLAB simulations were conducted to evaluate the performance of the MASH architecture.

4.2.1 Gain-Phase Model Simulation Results

With ideal integrators, the SNR is found to be 157 dB over a 200Hz bandwidth, a result better than that predicted in Chapter 3. This is because the calculation performed by *Model.m* uses the unity gain approximation. As mentioned in Chapter 1, the unity gain approximation is not a very good model of a 1-bit quantizer because it does not have many levels. Because the Gain-Phase Model does not attempt to exactly cancel the noise, however, the system still performs well, and indeed exceeds expectations.

The results of the capacitor mismatch simulations and finite op amp gain simulations are shown in Figures 4.3 and 4.4, respectively. Histograms of the capacitor mismatch Montecarlo simulations are included in Appendix D.

4.2.2 Simulation Results for Model with n=17

With ideal integrators, the SNR is found to be 160.8 dB over a 200Hz bandwidth. This result is drastically poorer than that predicted by *Model.m* (~190 dB). One reason for this has already been mentioned: the unity gain approximation is not a very good model of 1-bit quantizers. The n=17 Model attempts to precisely cancel an error which cannot even be accurately determined by the unity gain approximation. As a result, what seems in

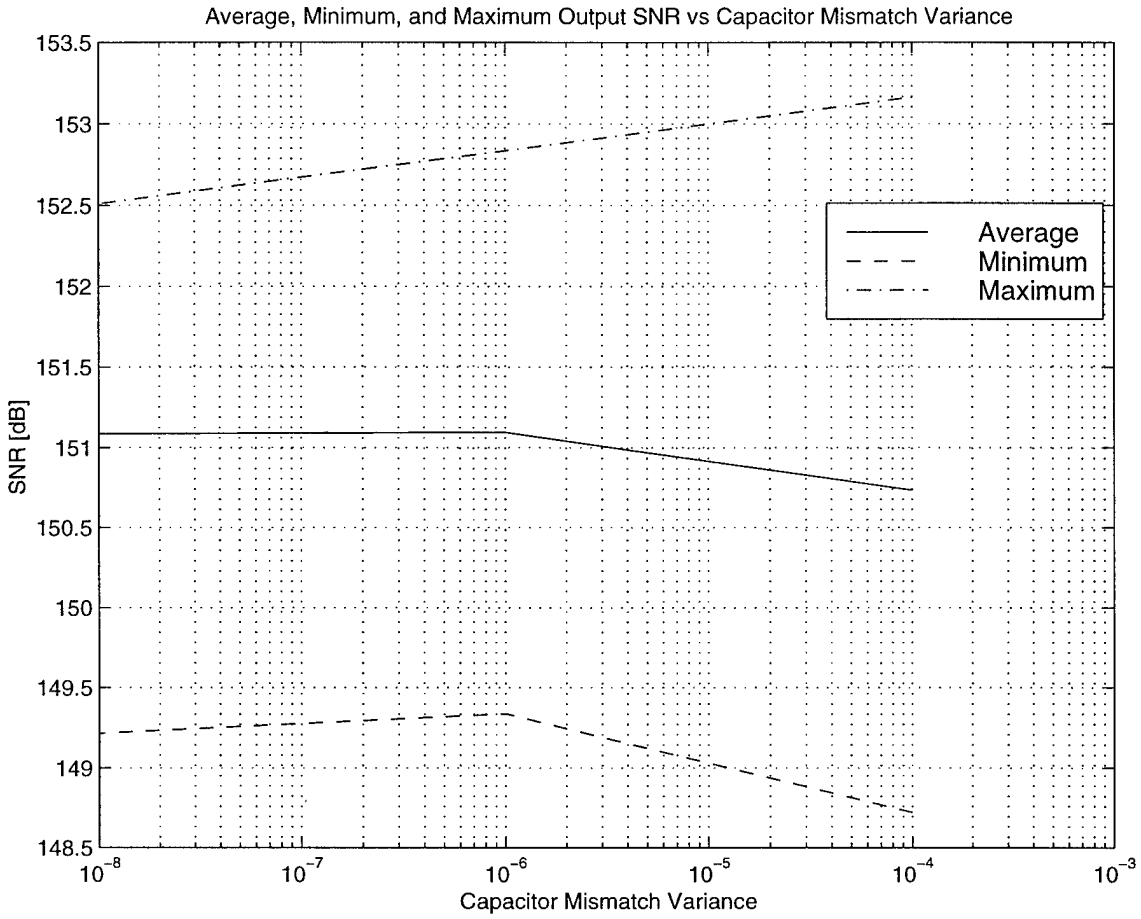

Figure 4.3: Capacitor Mismatch Simulation Results for MASH Architecture (Gain-Phase Model)

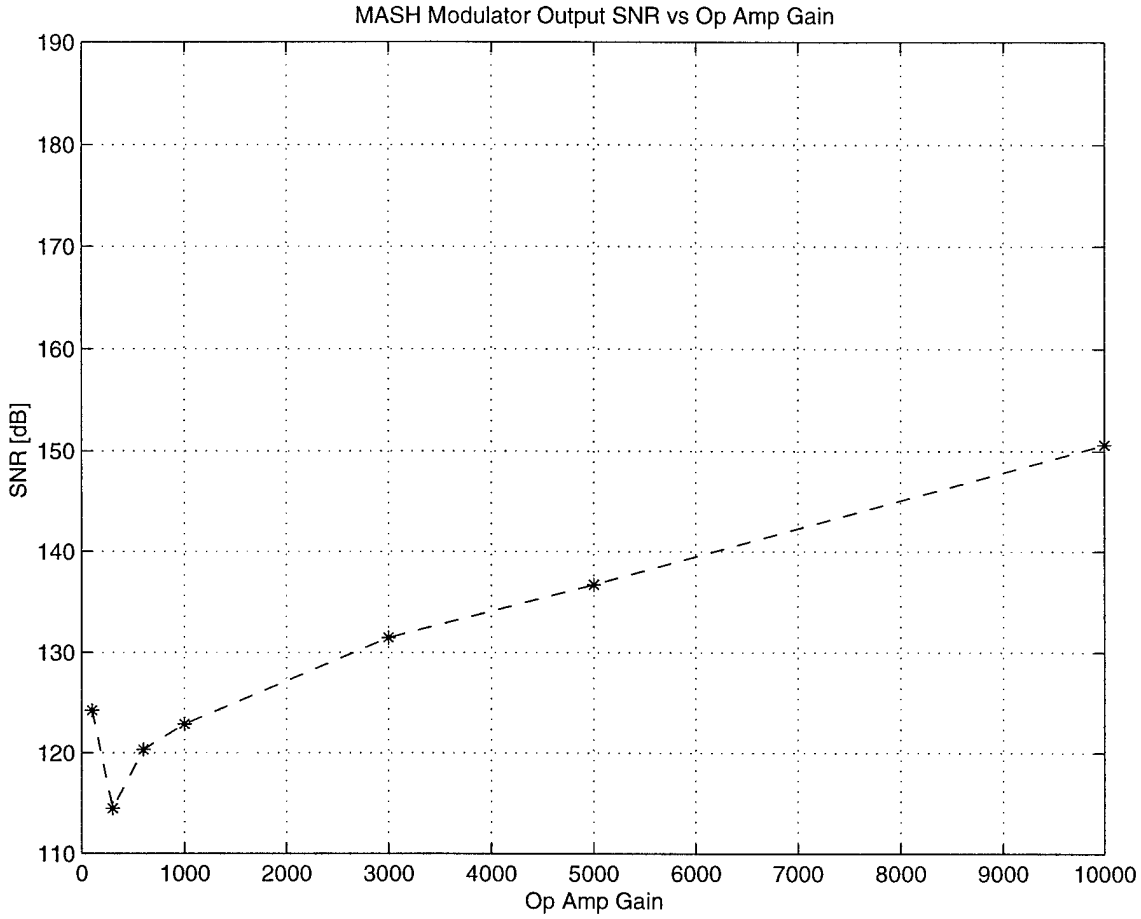

Figure 4.4: Plot of MASH Modulator (Gain-Phase Model) SNR
With Varying Op Amp Gain

theory to be a more accurate and precise error cancellation filter, is in fact, a much poorer design.

Capacitor mismatch montecarlo simulation results show that for a capacitor mismatch variance of 0.0001, the output SNR varies between 156.3 dB and 183.4 dB with an average SNR of 168.7 dB. The histogram giving this result is included in Appendix D.

The results of the op amp gain variation simulations are illustrated in Figure 4.5.

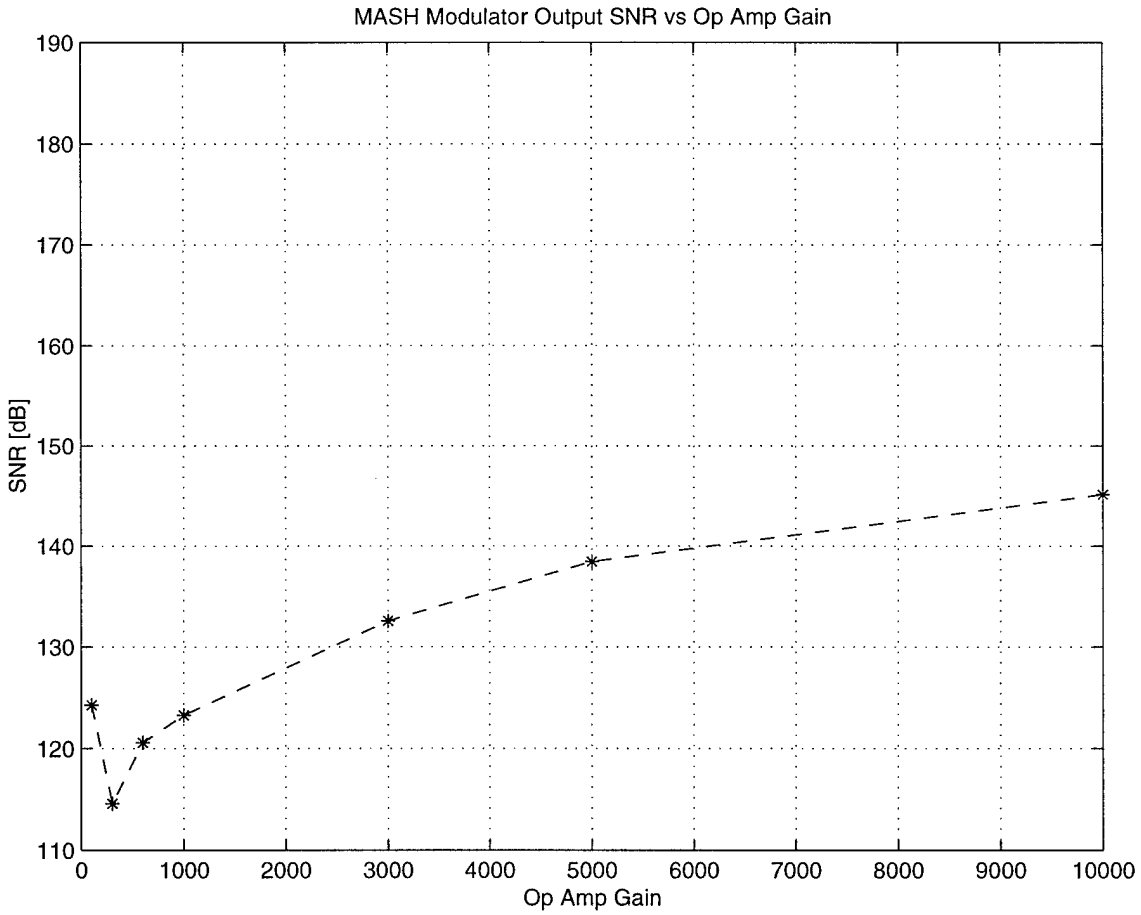

Figure 4.5: Plot of MASH Modulator (Model with n=17) SNR
With Varying Op Amp Gain

4.3 Summary of Results

Table 3 summarizes the results of the simulations presented above. For ease of comparison the predicted results computed in previous chapters are also included.

	Predicted SNR (dB)	Simulated SNR (dB)	SNR Variation Given Cap. Mismatch (dB)	SNR Variation With Finite Op Amp Gain (dB)
Single Loop Modulator	186	186	184 - 189	127 - 182
Gain-Phase MASH Modulator	149	157	148 - 153	114 - 151
n=17 Model MASH Modulator	189	160	156 - 183	114 - 145

Table 3: Summary of Predicted and Simulated Results

From these results it can be seen that in terms of SNR performance, the Single Loop Modulator clearly performs better than the MASH Modulator. While using more error cancellation filter coeffients in theory gives as high an SNR as the Single Loop Modulator, in actuality the limitations of the unity gain approximation cause the MASH modulator to yield relatively poorer SNR performance.

In terms of robustness, both architectures perform comparably. The Single Loop Modulator and Gain-Phase MASH Modulator vary by only 5 dB with capacitor mismatch variation. For op amp gains greater than roughly 8000 the effect of gain variation on SNR performance is also insignificant.

As expected from the analysis in Chapter 4, the n=17 Model MASH Modulator is much more sensitive to capacitor mismatch variations than the Gain-Phase MASH Modulator. The SNR performance is quite unstable, with a variation of 27 dB! Additionally, the n=17 Model MASH Modulator performance is also more negatively impacted by low op amp gain. With a gain of 10,000 the SNR is still 15 dB less than the ideal simulated SNR.

CHAPTER 5

CONCLUSION

The performance of two 4th order delta-sigma A/D topologies were analyzed: that of a single loop modulator, and that of a MASH modulator. MATLAB simulation results showed that given the same sampling frequencies, the single loop modulator yielded a higher SNR than the MASH modulator. Both architectures performed comparably with regards to robustness. However, MASH designs with higher order error cancellation filters are very sensitive to capacitor mismatch variations and finite op amp gain non-idealities.

The reason for the poorer SNR performance of the MASH architecture involves a fundamental design issue of MASH modulators: designing an error cancellation network that can accurately cancel the first stage quantizer error. Several possible cancellation filter design approaches were presented. However, because of the limitations of the unity gain approximation, error cancellation filters designed based on this approximation do not yield good results.

Once possible way of improving MASH architecture performance involves using a more accurate quantizer model based on describing functions. The design of delta-sigma modulators using model that combines describing function analysis, analytical modeling, and empirical approximation (Adaptive Gain Model) was presented by Louis Williams in his 1993 Doctoral Thesis [4]. In that work the Adaptive Gain Model was used to design a 3rd order 2-1 lowpass cascaded architecture. Use of such a model was

shown to yield results that allowed for the prediction of other effects such as the overload performance of the modulator. However, such an analysis is quite involved. More work needs to be done to further develop such methods.

Another method of improving MASH architecture performance involves using a multi-bit quantizer in the second-stage, and thereby better satisfy the requirements for the unity gain approximation. The key concern in using multi-bit quantizers, however, is linearity. While with 1-bit quantizers the quantizer is ensured to preserve the linearity of the system, the use of multi-bit quantizers introduces additional errors and distortion from the quantizer. Applications with stringent linearity requirements seem to preclude the use of multi-bit quantizers. However, if the first stage is designed with a 1-bit quantizer, and only the second stage uses a multi-bit quantizer, then performance can be achieved with minimal effects on linearity: because the input signal of the second stage is a signal on the order of the quantizer error (i.e. small amplitude), the distortion introduced by non-linearity in the second stage quantizer is correspondingly smaller.

In conclusion, while MASH architectures designed with the conventional unity gain approximation of the quantizer do not yield as good SNR performance as a single loop design, future work on the design of error cancellation filters and quantizer modeling may enable better and more comparable MASH modulator designs.

APPENDIX A

Delta-Sigma Toolbox for MATLAB

MATLAB's internal representation of the modulator is accomplished using the ABCD matrix. The ABCD matrix is essentially the state-space description of the modulator, and stores in matrix form the following equations:

$$x(n+1) = Ax(n) + B[u(n)\ v(n)]'$$

$$y(n) = Cx(n) + D[u(n)\ v(n)]'$$

where x is the vector of state variables, y is the input to the quantizer, u is the input of the modulator, and v is the output of the modulator (and quantizer).

Thus, the ABCD matrix becomes

ABCD = [A B ; C D] using MATLAB notation.

The following is a list of the Toolbox functions utilized in designing the modulator.

A.1 SynthesizeNTF

ntf = synthesizeNTF(order=3,R=64,opt=0,H_inf=1.5,f0=0)
Synthesize a noise transfer function for a delta-sigma modulator.

order = order of the modulator
R = oversampling ratio
opt = flag for optimized zeros
0 -> not optimized,
1 -> optimized,
2 -> optimized with at least one zero at band-center
H_inf = maximum NTF gain
f0 = center frequency (1->fs)

ntf is a zpk object containing the zeros and poles of the NTF. See zpk.m

A.2 PredictSNR

[snr,amp,k0,k1,sigma_e2] = predictSNR(ntf,R=64,amp=...,f0=0)
Predict the SNR curve of a binary delta-sigma modulator by using the describing function method of Ardalan and Paulos.

The modulator is specified by a noise transfer function (ntf).
The band of interest is defined by the oversampling ratio (R)
and the center frequency (f0).
The input signal is characterized by the amp vector.
amp defaults to [-120 -110...-20 -15 -10 -9 -8 ... 0]dB, where 0 dB means
a full-scale (peak value = 1) sine wave.

The algorithm assumes that the amp vector is sorted in increasing order;
once instability is detected, the remaining SNR values are set to -Inf.

Output:
snr a vector of SNR values (in dB)
amp a vector of amplitudes (in dB)
k0 the quantizer signal gain
k1 the quantizer noise gain
sigma_e2 the power of the quantizer noise (not in dB)

The describing function method of A&P assumes that the quantizer processes
signal and noise components separately. The quantizer is modelled as two
(not necessarily equal) linear gains, k0 and k1, and an additive white
gaussian noise source of power sigma_e2. k0, k1 and sigma_e2 are calculated
as functions of the input.
The modulator's loop filter is assumed to have nearly infinite gain at
the test frequency.

A.3 RealizeNTF

[a,g,b,c] = realizeNTF(ntf,form='CRFB',stf=1)
Convert a noise transfer function into coefficients for the desired structure.
Supported structures are

- CRFB Cascade of resonators, feedback form.
- CRFF Cascade of resonators, feedforward form.
- CIFB Cascade of integrators, feedback form.
- CIFF Cascade of integrators, feedforward form.

CRFBD CRFB with delaying quantizer.
See the accompanying documentation for block diagrams of each structure

The order of the NTF zeros must be (real, complex conj. pairs).
The order of the zeros is used when mapping the NTF onto the chosen topology.

A.4 StuffABCD

ABCD = stuffABCD(a,g,b,c,form='CRFB')
Compute the ABCD matrix for the specified structure.
See realizeNTF.m for a list of supported structures.
mapABCD is the inverse function.

A.5 MapABCD

[a,g,b,c] = mapABCD(ABCD,form='CRFB')
Compute the coefficients for the specified structure.
See realizeNTF.m for a list of supported structures.
stuffABCD is the inverse function.

A.6 ScaleABCD

[ABCDs,umax,S]=scaleABCD(ABCD,nlev=2,f=0,xlim=1,ymax=nlev+5,umax,N=1e5)
Scale the loop filter of a general delta-sigma modulator for dynamic range.

ABCD The state-space description of the loop filter.
nlev The number of levels in the quantizer.
xlim A vector or scalar specifying the limit for each state variable.
ymax The stability threshold. Inputs that yield quantizer inputs above ymax
are considered to be beyond the stable range of the modulator.
umax The maximum allowable input amplitude. umax is calculated if it
is not supplied.

ABCDs The state-space description of the scaled loop filter.
S The diagonal scaling matrix, ABCDs = [S*A*Sinv S*B; C*Sinv D];
xs = S*x;

A.7 CalculateTF

[ntf,stf] = calculateTF(ABCD,k=1)
Calculate the NTF and STF of a delta-sigma modulator whose loop filter
is described by the ABCD matrix, assuming a quantizer gain of k.
The NTF and STF are zpk objects.

APPENDIX B

MATLAB CODE FOR SINGLE LOOP MODULATOR SIMULATIONS

B.1 Code for Calculating Expected SNR

```
%%%%% PREDICT4.M %%%%%

N=16*64*2048;
fs=2*200*3200;
time=[1:N]/fs;
fin=20000;
in=1;
u=in*cos(2*pi*fin*time);
e1=2.5*(rand(1,N)-0.5);

stfn=[1];
stfd=[1];
ntfn=conv([1 -1.99 1],[1 -1.99 1]);
ntfd=conv([1 -1.491 0.5652],[1 -1.687 0.7865]);

a1=0.3964;
a2=0.3754;
a3=0.4079;
a4=0.3890;
g1=0.1116;
g2=0.0227;
b1=0.3964;
b2=0.3754;
b3=0.4079;
b4=0.3890;
b5=1;
c1=0.0863;
c2=0.2361;
c3=0.4242;
c4=1.4277;

newntfn=[1 (g1*c1+g2*c3-4) (g1*c1-2)*(g2*c3-2)+2
(g1*c1+g2*c3-4) 1];
const1=g1*c1+g2*c3-4+c4*(a4+a3*c3);
const2=2+(g1*c1-2)*(g2*c3-2)+c4*(c2*c3*(a2+a1*c1)+(g1*c1-
2)*(a4+a3*c3)-a4);
```

```
const3=g1*c1+g2*c3-4+c4*(-c2*c3*a2-a4*(g1*c1-2)+a4+a3*c3);
const4=1-c4*a4;
newntfd=[1 const1 const2 const3 const4];

sig11=filter(stfn,stfd,u);
sig12=filter(newntfn,newntfd,e1);
Y=sig11+sig12;

SIGNAL=Y;
CalcSNR;
```

B.2 Capacitor Mismatch Monte Carlo Simulation Code

```
%%%%%%%% MONTECARLO.M %%%%%%%%

for i=1:100,

A=10^6;
C2=1;
C1=1+0.003*randn(1,1);
n1=C1*A/(C2*(1+A));
n2=(C1+(1+A)*C2)/(C2*(1+A));
n3=n1;
n4=n2;
n5=n1;
n6=n2;
n7=n1;
n8=n2;

Fourth
SIGNAL=out;
CalcSNR;
DATA1(i)=SNR

end;
```

B.3 Op Amp Gain Variation Simulation Code

```
%%%%%%%% GAINVAR.M %%%%%%%%

Gain=[100 300 600 1000 3000 5000];

for ind=1:length(Gain),

A=Gain(ind);
C2=1;
C1=1;
n1=C1*A/(C2*(1+A));
```

```
n2=(C1+(1+A)*C2)/(C2*(1+A));
n3=n1;
n4=n2;
n5=n1;
n6=n2;
n7=n1;
n8=n2;

Fourth
SIGNAL=out;
CalcSNR;
DATA1(ind)=SNR

end;
```

B.4 Calculation of Output of 4th Order Single Loop

```
%%%%%%%% FOURTH.M %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Implements the iterative calculation of the output
%  given the input.
%  Also takes into account non-idealities of integrators
%  Integrator1=n1z/(n2z-1)
%  Integrator2=n3/(n4z-1)
%  Integrator3=n5z/(n6z-1)
%  Integrator4=n7/(n8z-1)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


ABCDsn = [1.0000   -0.1116         0         0    0.3964   -
0.3964;
          0.0863    0.9904         0         0    0.4096   -
0.4096;
          0         0.2361    1.0000   -0.0227    0.4079   -
0.4079;
          0         0.1001    0.4242    0.9904    0.5620   -
0.5620;
          0              0         0    1.4277    1.0000
0];


a1=0.3964;
a2=0.3754;
a3=0.4079;
a4=0.3890;
g1=0.1116;
g2=0.0227;
b1=0.3964;
b2=0.3754;
b3=0.4079;
```

```
b4=0.3890;
b5=1;
c1=0.0863;
c2=0.2361;
c3=0.4242;
c4=1.4277;

ABCDs=[1/n2  -n1/n2*g1    0     0    n1/n2*b1       -
n1/n2*a1;
       n3/(n1*n4)*c1  1/n4-n1*n3/(n2*n4)*c1*g1    0     0
n3/n4*(b2+n1/n2*
b1*c1) -n3/n4*(a2+n1/n2*a1*c1);
      0             n5/n6*c2             1/n6           -
n5/n6*g2
 n5/n6*b3               -n5/n6*a3;
      0             n5*n7/(n6*n8)*c2*c3  n7/(n6*n8)*c3 1/n8-
n5*n7/(n6*n8)*g2*
c3  n7/n8*(b4+n5/n6*b3*c3) -n7/n8*(a4+n5/n6*a3*c3);
      0             0                    0              c4
 b5                     0];

ABCD=ABCDs;

OSR=3200;
fs=2*200*OSR;
N=16*64*2048;
in=1;
fin=20000;
time=[1:N]/fs;
noise_in=0.0000001;

u=in*cos(2*pi*fin*time)+randn(1,N)*noise_in;

dcoff=-1e-3;            % Integrator DC Offset
noise_ref=0.001;        % Ref thermal noise
noise_pref=0.001;
noise_nref=0.001;
n=0;
z=0;
x=0;
t=zeros(1,N);
out=zeros(1,N);
int1=zeros(1,N);
int2=zeros(1,N);
int3=zeros(1,N);
int4=zeros(1,N);
v=zeros(1,N);
f=zeros(1,N);

b5=1;
c1=0.0863;
c2=0.2361;
c3=0.4242;
```

```
c4=1.4277;

flag=0;

for n=2:N-1


% Comparator samples last integrator output first.

v(n)=ABCD(5,1)*int1(n)+ABCD(5,2)*int2(n)+ABCD(5,3)*int3(n)+A
BCD(5,4)*int4(n)+
ABCD(5,5)*u(n)+ABCD(5,6)*f(n);

Quant2;


% First Accumulator Stage

z=dcoff+ABCD(1,1)*int1(n)+ABCD(1,2)*int2(n)+ABCD(1,3)*int3(n
)+ABCD(1,4)*int4(
n)+ABCD(1,5)*u(n)+ABCD(1,6)*f(n);

if z>=2.5;
   disp('ouch1');
%  z=2.5;
   flag=1;
elseif z<-2.5
   disp('ouch1');
%  z=-2.5;
   flag=1;
end

int1(n+1)=z;

% Second Accumulator Stage

z=-
dcoff+ABCD(2,1)*int1(n)+ABCD(2,2)*int2(n)+ABCD(2,3)*int3(n)+
ABCD(2,4)*int4
(n)+ABCD(2,5)*u(n)+ABCD(2,6)*f(n);

if z>=2.5;
   disp('ouch2');
%  z=2.5;
   flag=1;
elseif z<-2.5
   disp('ouch2');
%  z=-2.5;
   flag=1;
end

int2(n+1)=z;
```

```
% Third Accumulator Stage

z=dcoff+ABCD(3,1)*int1(n)+ABCD(3,2)*int2(n)+ABCD(3,3)*int3(n
)+ABCD(3,4)*int4(
n)+ABCD(3,5)*u(n)+ABCD(3,6)*f(n);

if z>=2.5;
   disp('ouch3');
%  z=2.5;
   flag=1;
elseif z<-2.5
   disp('ouch3');
%  z=-2.5;
   flag=1;
end

int3(n+1)=z;

% Fourth Accumulator Stage

z=-
dcoff+ABCD(4,1)*int1(n)+ABCD(4,2)*int2(n)+ABCD(4,3)*int3(n)+
ABCD(4,4)*int4
(n)+ABCD(4,5)*u(n)+ABCD(4,6)*f(n);

if z>=2.5;
   disp('ouch4');
%  z=2.5;
   flag=1;
elseif z<-2.5
   disp('ouch4');
%  z=-2.5;
   flag=1;
end

int4(n+1)=z;

end
out=f;
```

B.5 Quantizer Code

```
%%%%% QUANT2.M %%%%%

   if v(n)>0.0;
                f(n)=2.50+randn(1,1)*noise_pref;
        end
        if v(n)<=0.0;
```

```
                f(n)=-2.50+randn(1,1)*noise_nref;
    end;
```

B.6 SNR Calculation Code

```
%  SNR Calculation

% Script expects input signal in SIGNAL, as well as fs and N
defined

disp('EXECUTE CalcSNR.m');

u=in*cos(2*pi*fin*time);
spec=fft(u.*blackman(length(u))');
P=10*log10((spec.*conj(spec)));
NP=max(P(1:round(N*2*fin/fs+1000)));


spec=fft(SIGNAL.*blackman(length(SIGNAL))');
P=10*log10((spec.*conj(spec)));
%P=P-max(P(1:round(N*2*fin/fs+1000)));
P=P-NP;
f=[1:N]*(fs/N);
figure(1);
plot(f(1:N/2),P(1:N/2));
grid;
title('Sigma-Delta A2D Output Spectrum for 20kHz FS/2');
xlabel('Frequency [Hz]');
ylabel('Amplitude [Hz]');
axis([fin-1000 fin+1000 -300 0]);

disp('Output Spectrum Calculation...Complete');

NBW=10*log10(1.73*fs/N);
N1=round((fin-30)*N/fs);
N2=round((fin-100)*N/fs);
N3=round((fin+30)*N/fs);
N4=round((fin+100)*N/fs);
tmp=[P(N1:N2) P(N3:N4)];
SNR=-(mean(tmp))-NBW;

disp('SNR Calculation...Complete');

SNR
```

APPENDIX C

MATLAB CODE FOR MASH MODULATOR SIMULATIONS

C.1 MATLAB Simulation Using Unity Gain Model

```
%%%%% MODEL.M %%%%%

N=16*64*2048;
fs=2*200*3200;
time=[1:N]/fs;
fin=20000;
in=1;
u=in*cos(2*pi*fin*time);
e1=2.5*(rand(1,N)-0.5);
e2=e1;

lambda=1;
beta=0.8;
BandErrCan;

sig11=filter(stfn,stfd,u);
sig12=filter(ntfn,ntfd,e1);
Y1=sig11+sig12;

u2=beta*((lambda-1)*Y1+e1);

sig21=filter(stfn,stfd,u2);
sig22=filter(ntfn,ntfd,e2);
Y2=sig21+sig22;

% overall output
HY1=filter(H1,[1],Y1);
HY2=filter(H2,[1],Y2);
OUT=HY1+HY2;
```

C.2 Capacitor Mismatch Monte Carlo Simulation Code

```
%%%%%%%% MONTEC.M %%%%%%%%

for ind=1:100,

A=10^6;
C2=1;
C1=1+0.0001*randn(1,1);
n1=C1*A/(C2*(1+A));
n2=(C1+(1+A)*C2)/(C2*(1+A));
n3=n1;
n4=n2;

OptDes
SIGNAL=OUT;
CalcSNR;
DATA2(ind)=SNR

end;
```

C.3 Op Amp Gain Variation Simulation Code

```
%%%%%%%% GAINC.M %%%%%%%%

Gain=[100 300 600 1000 3000 5000];

for ind=1:length(Gain),

A=Gain(ind);
C2=1;
C1=1;
n1=C1*A/(C2*(1+A));
n2=(C1+(1+A)*C2)/(C2*(1+A));
n3=n1;
n4=n2;

OptDes
SIGNAL=OUT;
CalcSNR;
DATA1(ind)=SNR
end;
```

C.4 Error Cancellation Filter Calculation Code

```
%%%%% BANDERRCAN.M %%%%%

% BANDPASS DELTA-SIGMA ERROR CANCELLATION FILTER CALCULATION
% Requires G, beta and lambda defined prior to running

G=1;

% System Block Diagram Scaling Coefficient Values
a1=0.4846;
a2=0.1261;
b1=0.4846;
b2=0.1261;
b3=1;
c1=0.0992;
c2=4.4286;
g1=0.0971;

% NTF and STF Coefficients
k1=G*b3;
k2=G*(b3*(g1*c1-2)+c2*(b1*c1+b2));
k3=G*(b3-b2*c2);
k4=g1*c1-2+G*c2*(a1*c1+a2);
k5=1-G*a2*c2;
k6=g1*c1-2;
stfn=[k1 k2 k3];
stfd=[1 k4 k5];
ntfn=[1 k6 1];
ntfd=stfd;

% Calculate Error Cancellation Filters
% Full Precision Exact Noise Cancellation Filters
% H1=-beta*[lambda k4+(lambda-1)*k6 k5+lambda-1];
% H2=[1 k6 1];

% Approximate Noise Cancellation - 2nd Order H2
K=4.5
H1=[0 1 0 0];
H2=-K/beta*[0 1 k6 1];

% Approximate Noise Cancellation - More Accurate Method
% M=17;
% denom=[lambda   k4+(lambda-1)*k6   k5+lambda-1];
% sys=tf([1 0 0],denom,-1);
% fir_resp=impulse(sys)';
% H1=[0 1 zeros(1,M+1)];
% H2=-(1/beta)*conv([0 1 k6 1],fir_resp(1:M));

EC1=conv(ntfn,(H1+beta*(lambda-1)*H2));
EC2=beta*conv(ntfd,H2);
ERROR_COEFF=EC1+EC2
```

C.5 MASH Architecture Simulation Code

```
%%%%% OPTDES.M %%%%%

LambdaVector=[1];
BetaVector=[0.8];

G=1;
FLAG=0;
OPTL=0;
OPTB=0;
OPTSNR=0;

for i=1:length(LambdaVector)
    for j=1:length(BetaVector)
        disp('NEW RUN');
        lambda=LambdaVector(i)
        beta=BetaVector(j)
        MyDes;
        if FLAG==1,
           FLAG=0;
        end;

% Error Cancellation
BandErrCan

% overall output
HY1=filter(H1,[1],Y1);
HY2=filter(H2,[1],Y2);
OUT=HY1+HY2;

        if FLAG==0,
           SIGNAL=OUT;
           CalcSNR;
           SNR
           if SNR > OPTSNR,
              OPTSNR=SNR;
              OPTL=lambda;
              OPTB=beta;
            end;
        end;
end;
end;
```

C.6 Subroutine To Simulate System Until Error Cancellation

```
%%%%% MYDES.M %%%%%

ABCDs=[1 -0.0971 0.4846 -0.4846; 0.0992 0.9904 0.1742 -
0.1742; 0 4.4286 1 0];

a1=0.4846;
a2=0.1261;
g1=0.0971;
b1=0.4846;
b2=0.1261;
b3=1;
c1=0.0992;
c2=4.4286;

ABCDs=[1/n2 -n1/n2*g1 n1/n2*b1 -n1/n2*a1; n3/n2/n4*c1
(1/n4-n1/n2/n4*c1*g1)
(n3/n4*b2+n1*n3/n2/n4*b1*c1)  -(n3/n4*a2+n1*n3/n2/n4*a1*c1);
0 c2 b3 0];

ABCD=ABCDs;

OSR=3200;
fs=2*200*OSR;
N=16*64*2048;
in=2;
fin=20000;
time=[1:N]/fs;
noise_in=0.0000001;

u=in*cos(2*pi*fin*time)+randn(1,N)*noise_in;
% u=randn(1,N)*noise_in;
MyDes2;
Y1=out;
V1=v;
if flag==1,
  FLAG=1;
end;

u=beta*(lambda*Y1-V1);
MyDes2;
Y2=out;
V2=v;
if flag==1;
  FLAG=1;
end;
```

C.7 Calculation of Output of 2nd Order Single Loop Stage

```
%%%%% MYDES2.M %%%%%


disp('SECOND ORDER SECTION');


dcoff1=-1e-3;   % 1st Integrator DC Offset
dcoff2=1e-3;    % 2nd Integrator DC Offset
noise_pref=0.001;       % Pref thermal noise
noise_nref=0.001;       % Nref thermal noise
n=0;
z=0;
x=0;
t=zeros(1,N);
out=zeros(1,N);
int1=zeros(1,N);
int2=zeros(1,N);
v=zeros(1,N);
f=zeros(1,N);

flag=0;

for n=2:N-1


      % Comparator samples last integrator output first.

v(n)=ABCD(3,1)*int1(n)+ABCD(3,2)*int2(n)+ABCD(3,3)*u(n)+ABCD
(3,4)*f(n);

   Quant2;
%   Quantizer
% f(n)=v(n)+2.5*(rand(1,1)-0.5)/(2^4);

% First Accumulator Stage


z=dcoff1+ABCD(1,1)*int1(n)+ABCD(1,2)*int2(n)+ABCD(1,3)*u(n)+
ABCD(1,4)*
f(n);
        if z>=2.5;
      disp('ouch');
%      z=2.5;
      flag=1;
   elseif z<-2.5
      disp('ouch');
%               z=-2.5;
       flag=1;
        end
```

```
        int1(n+1)=z;

% Second Accumulator Stage


x=dcoff2+ABCD(2,1)*int1(n)+ABCD(2,2)*int2(n)+ABCD(2,3)*u(n)+
ABCD(2,4)*
f(n);

if x>=2.5;
   disp('ouch');
%                x=2.5; flag=1;
   elseif x<-2.5
      disp('ouch');
%                x=-2.5; flag=1;
        end

   int2(n+1)=x;

end
out=f;
```

APPENDIX D

HISTOGRAMS FROM CAPACITOR MISMATCH SIMULATIONS

D.1 Single Loop Architecture

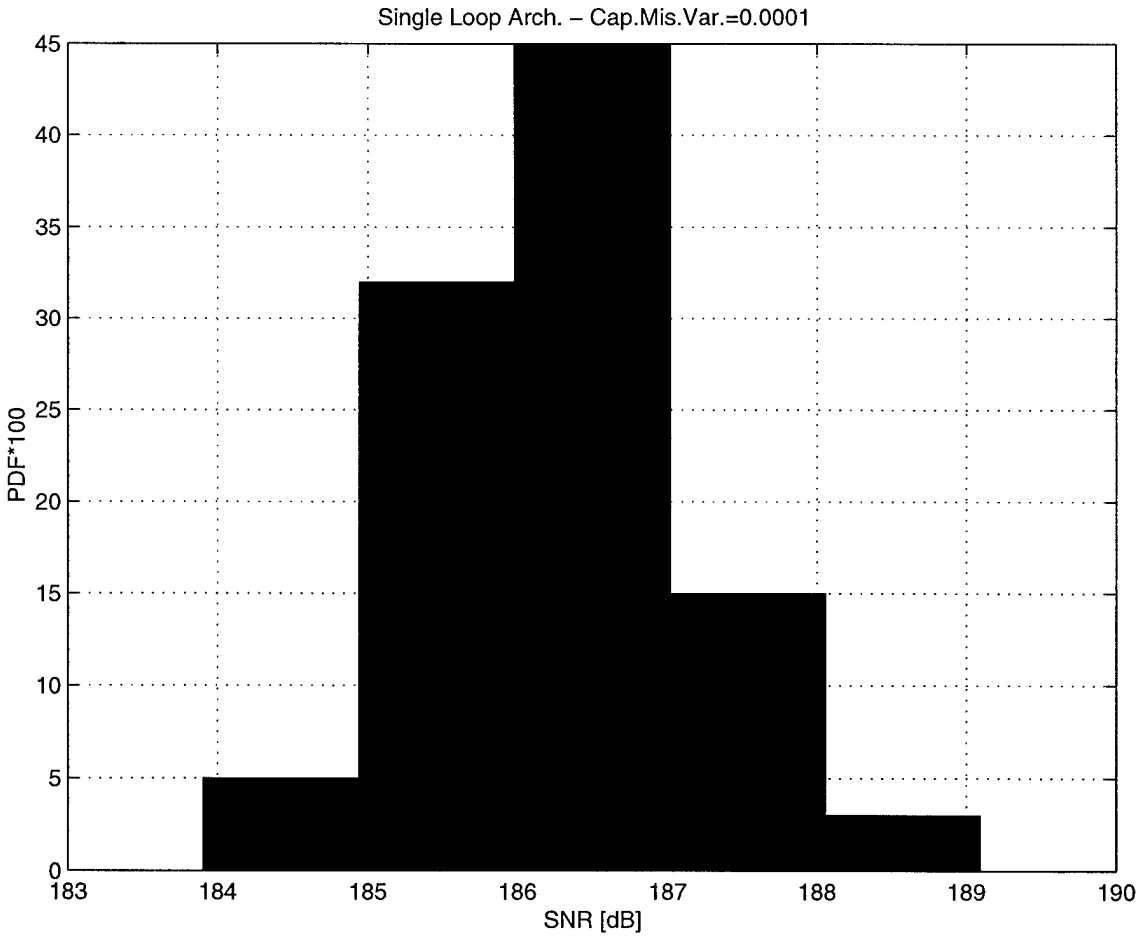

Figure D.1: Single Loop Architecture
Capacitor Mismatch Histogram - Variance of 10^{-4}

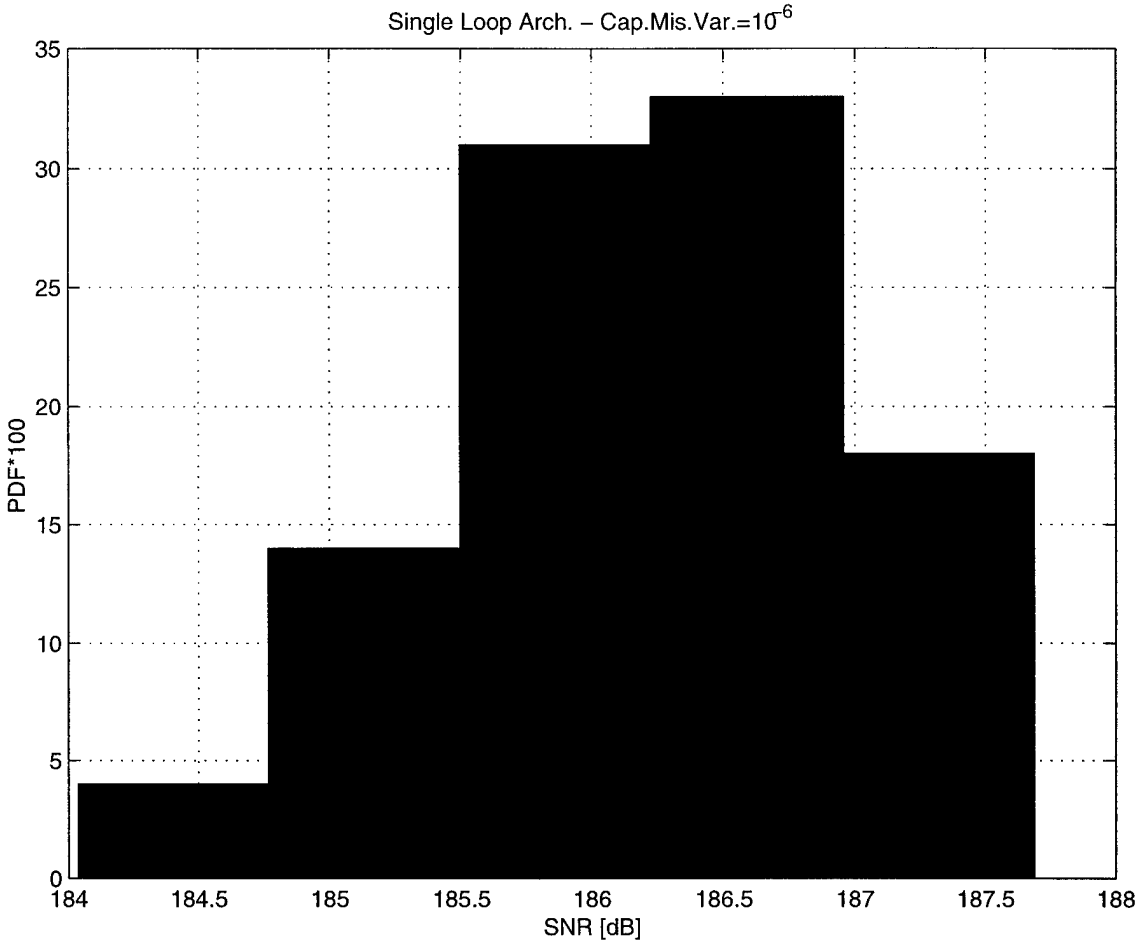

Figure D.2: Single Loop Architecture
Capacitor Mismatch Histogram - Variance of 10^{-6}

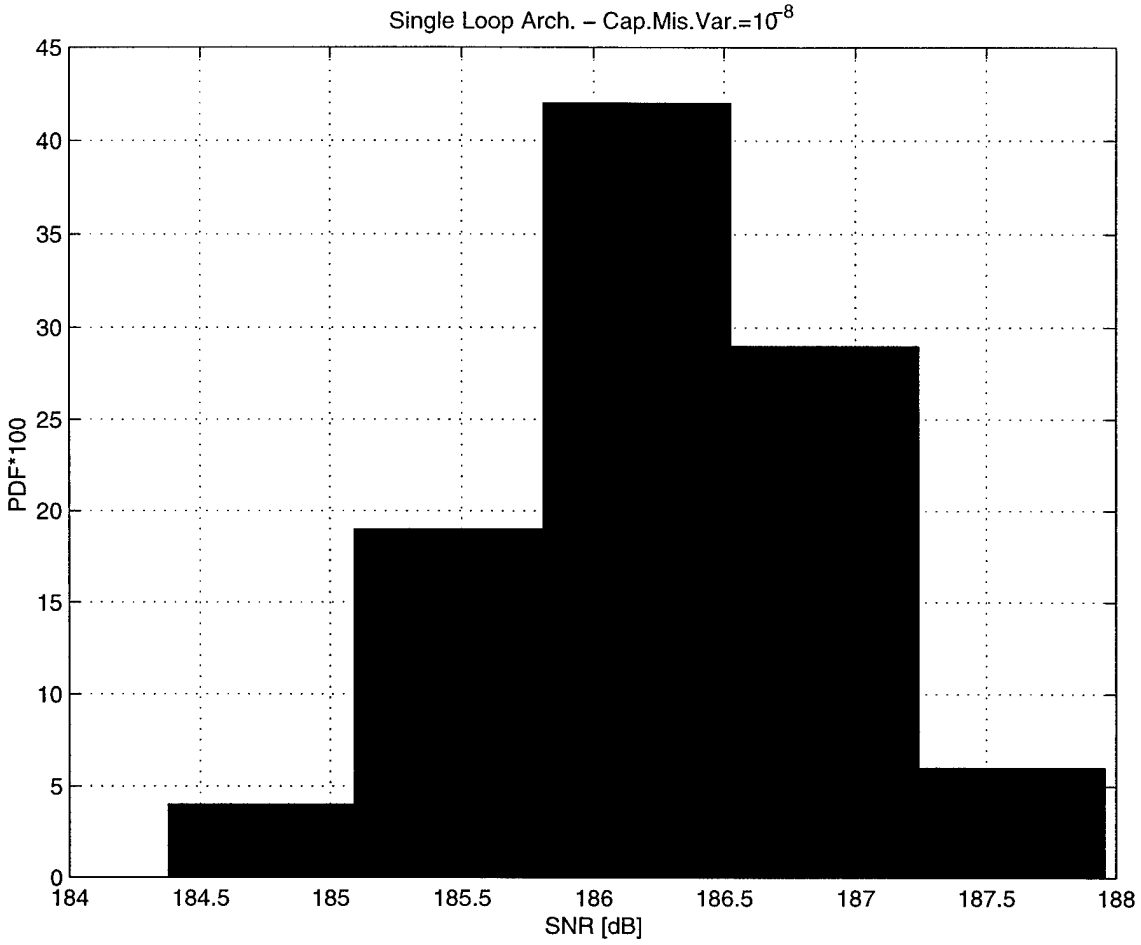

Figure D.3: Single Loop Architecture
Capacitor Mismatch Histogram - Variance of 10^{-8}

D.2 MASH Architecture (Gain-Phase Model)

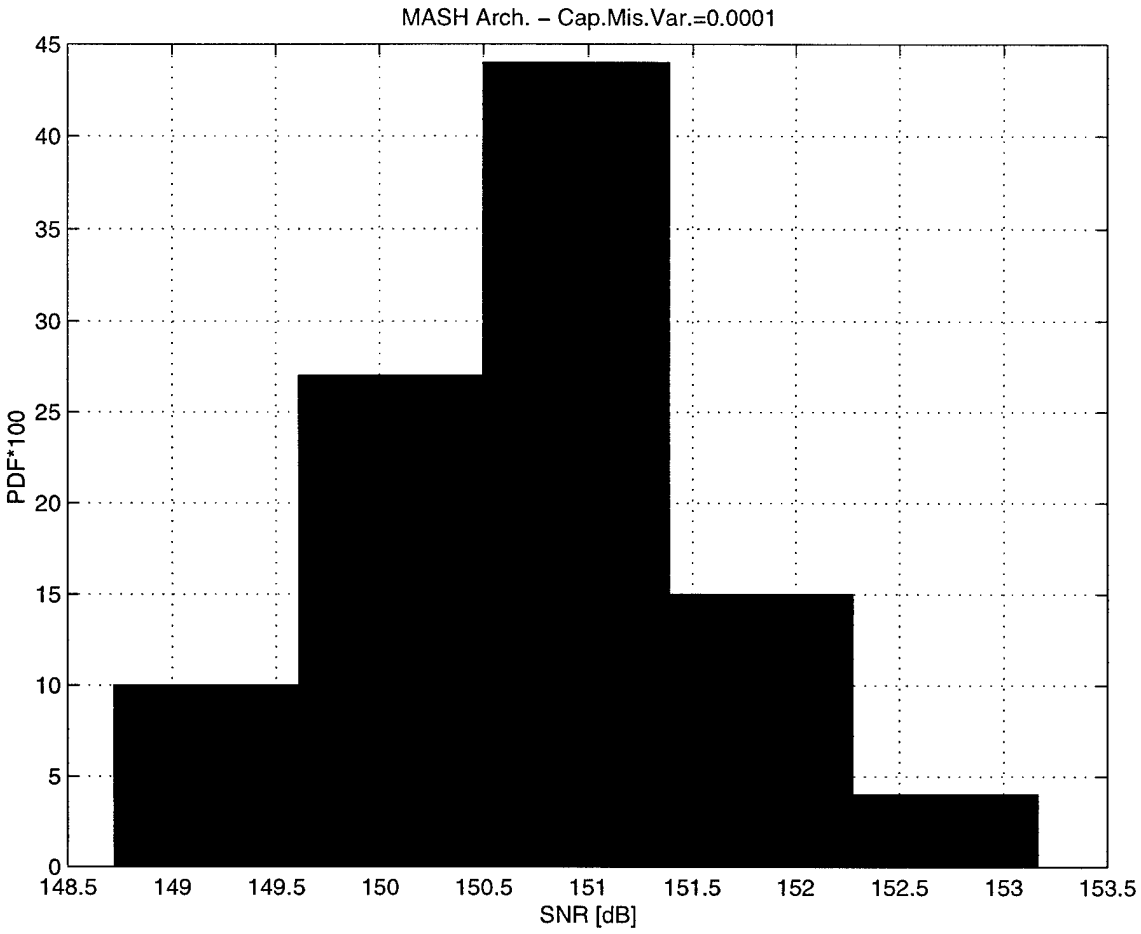

Figure D.4: MASH Architecture (Gain-Phase Model)
Capacitor Mismatch Histogram - Variance of 10^{-4}

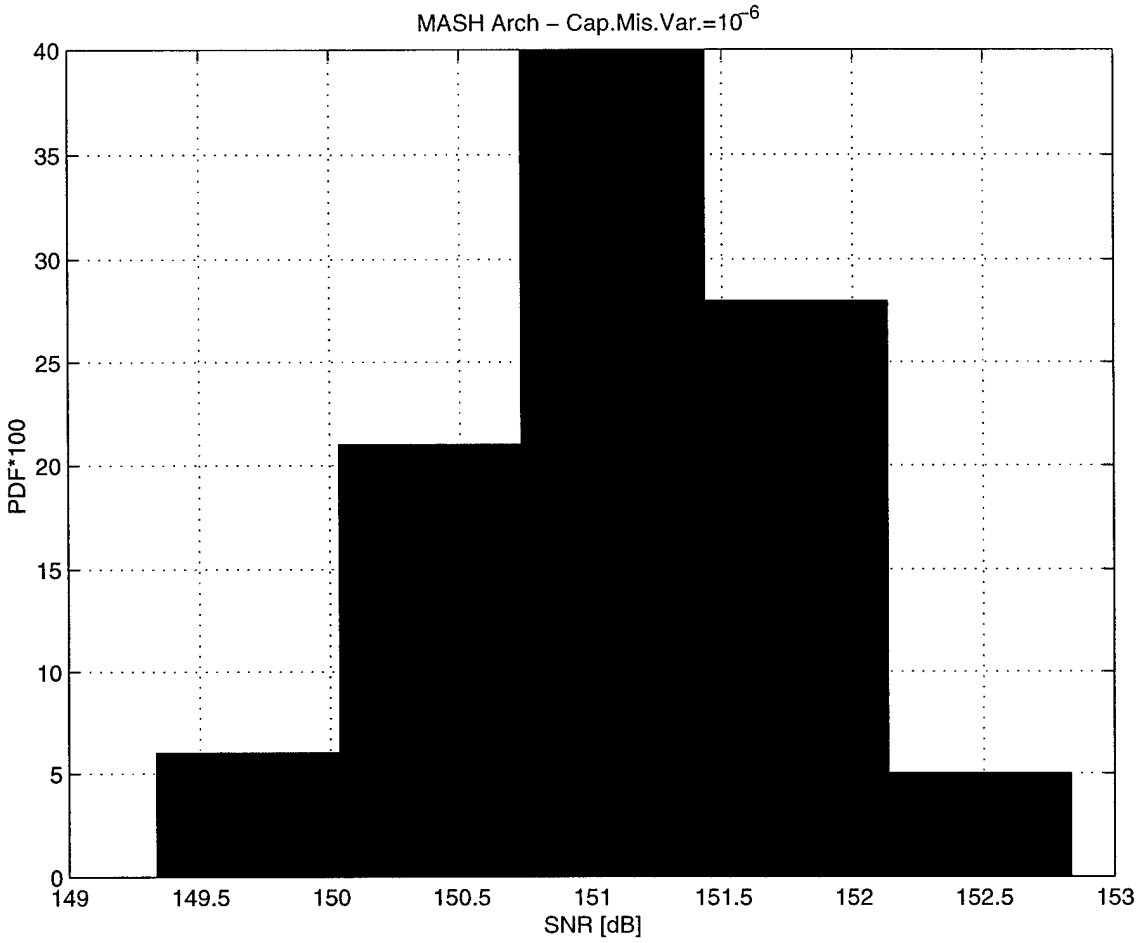

Figure D.5: MASH Architecture (Gain-Phase Model)
Capacitor Mismatch Histogram - Variance of 10^{-6}

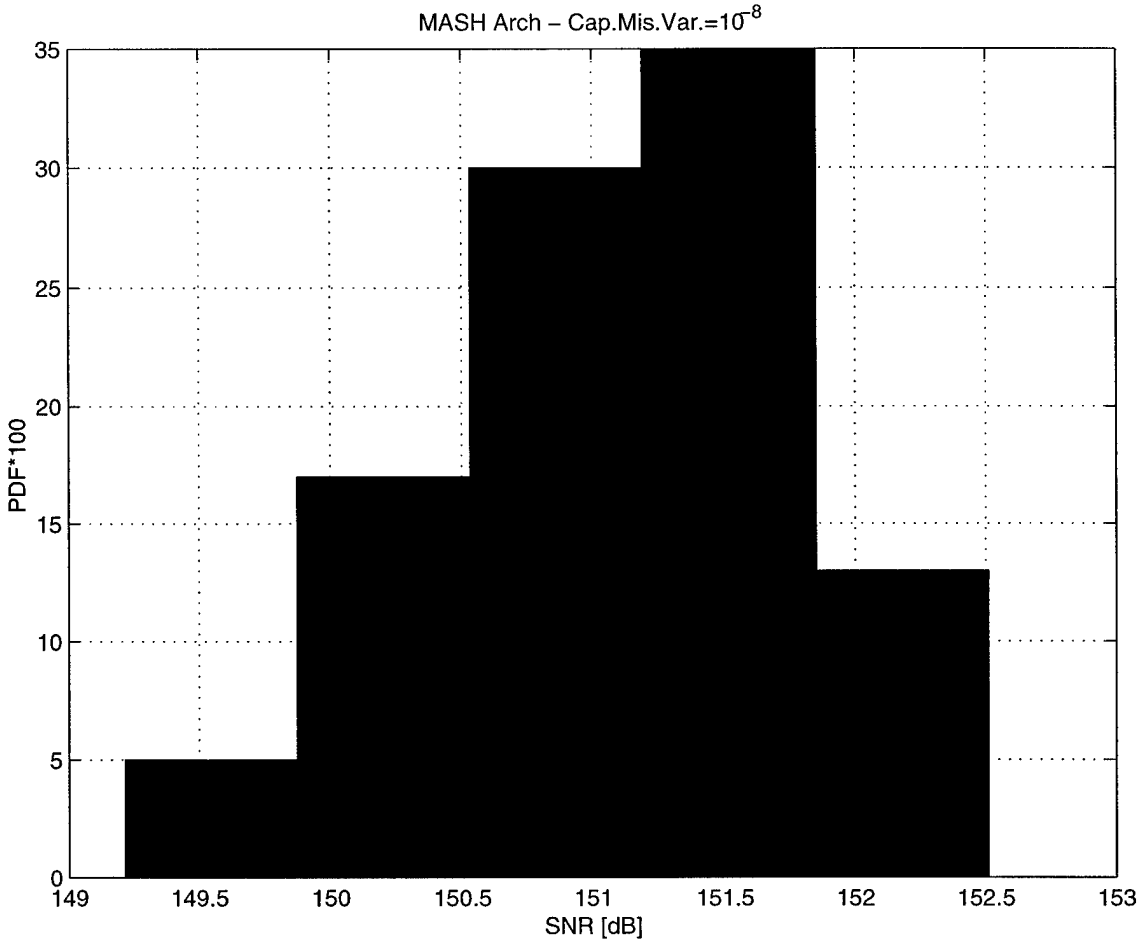

Figure D.6: MASH Architecture (Gain-Phase Model)
Capacitor Mismatch Histogram - Variance of 10^{-8}

D.3 MASH Architecture (Model with n=17)

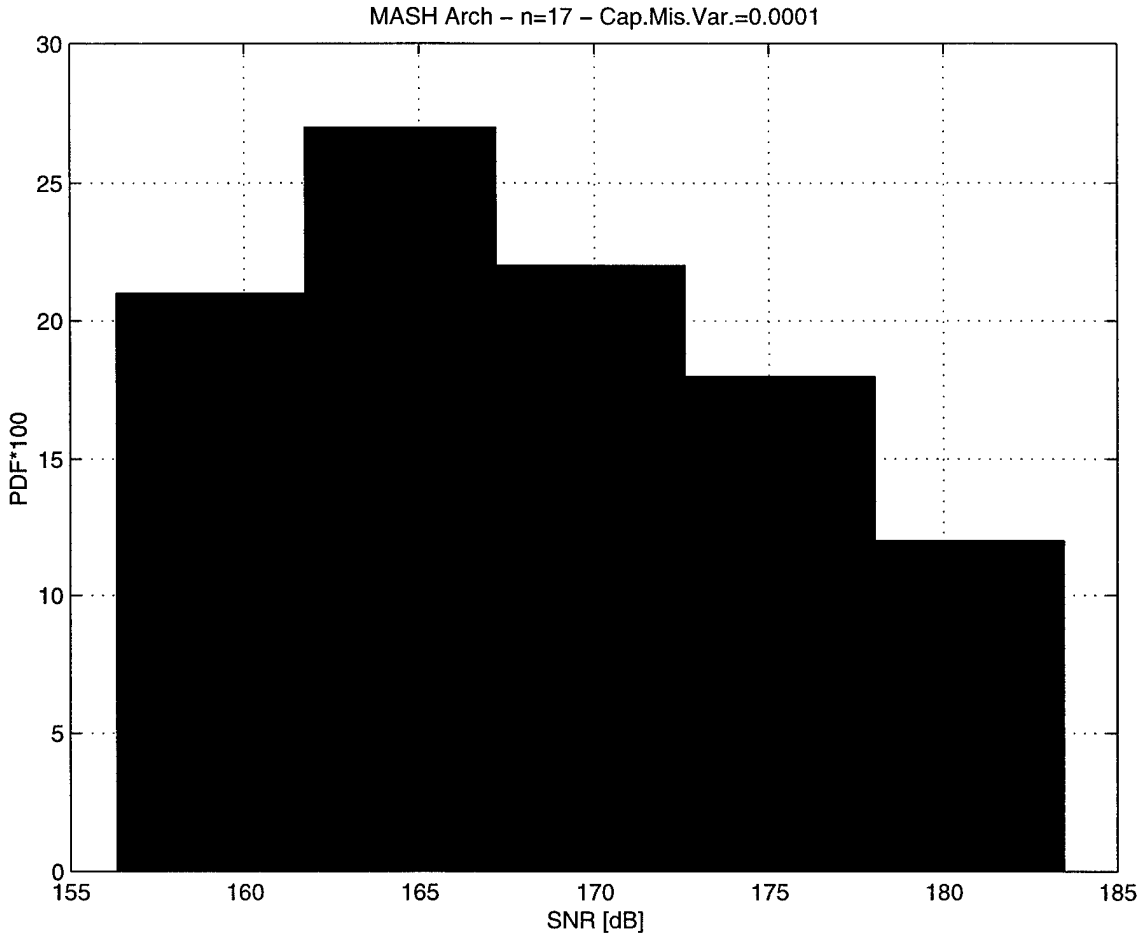

Figure D.7: MASH Architecture (Model with n=17)
Capacitor Mismatch Histogram - Variance of 10^{-4}

BIBLIOGRAPHY

[1] J. Candy and G. Temes, "Oversampling Methods for A/D and D/A Conversion." In *Oversampling Delta-Sigma Converters*, pp. 1-25, IEEE Press, 1992.

[2] P. Aziz, H. Sorensen and J. van der Spiegel, "An Overview of Sigma-Delta Converters." IEEE Signal Processing Magazine, pp. 61-84, January 1996.

[3] M. Hauser, "Principles of Oversampling A/D Conversion." Journal of the Audio Engineering Society, Vol. 39, No. 1/2, pp. 3-26, January/February 1991.

[4] L. Williams, "Modelling and Design of High-Resolution Sigma-Delta Modulators." Stanford University, Department of Electrical Engineering, Doctoral Thesis, August 1993.

[5] W. Bennett, "Spectra of Quantized Signals." Bell System Tech. Journal, Vol. 27, pp. 446-472, 1948.

[6] B. Widrow, "A Study of Rough Amplitude Quantization by Means of Nyquist Sampling Theory." IRE Trans. Circuit Theory, Vol. CT-3, pp. 266-276, December 1956.

[7] S. Ardalan and J. Paulos, "An Analysis of Nonlinear Behavior in Delta-Sigma Modulators." IEEE Transactions on Circuits and Systems, Vol. CAS-34, No. 6, June 1987.

[8] S. Nadeem, "A 16-Channel Oversampled Analog-to-Digital Converter for Multichannel Applications." Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Doctoral Thesis, August 1993.

[9] K. Chao, S. Nadeem, W. Lee, and C. Sodini, "A Higher Order Topology for Interpolative Modulators for Oversampling A/D Converters." IEEE Transactions on Circuits and Systems, Vol. 37, No. 3, March 1990.

[10] D. Ribner, "Multistage Bandpass Delta Sigma Modulators." IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, Vol. 41, No. 6, pp. 402-405, June 1994.

[11] M. Al-Janabi, I. Kale, and R. Morling, "MASH Structures for Bandpass Sigma-Delta Modulators." IEE Colloquium on Oversampling and Sigma-Delta Strategies for Digital Signal Processing, pp. 3/1-3/6, London, November 1995.

[12] M. Al-Janabi, I. Kale, and R. Morling, "Tunable Centre Frequency Resonator Based Bandpass Σ–Δ Modulators." IEEE Instrumentation and Measurement Technology Conference, pp. 1248-1252, St. Paul, Minnesota, USA, May 18-21, 1998.

[13] D. Morche, A. Aubert, E. Andre, F. Balestro, and P. Senn. "A New Multistage Bandpass Sigma-Delta Modulator." 1996 IEEE International Symposium on Circuits and Systems. Circuits and Systems Connecting the World, ISCAS 96, Vol. 1, pp. 13-16, New York, NY, USA, 1996.

[14] M. Al-Janabi, I. Kale, and R. Morling, "Effective-Fourth Order Resonator Based MASH Bandpass Sigma-Delta Modulators." 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing. IEEE Comput. Soc. Press. Vol.3 , pp. 2493-6, Los Alamitos, CA, USA, 1997.

[15] S. Jantzi, W. Snelgrove, and P. Ferguson, "A Fourth-Order Bandpass Sigma-Delta Modulator." IEEE Journal of Solid State Circuits, Vol. 28, No.3, pp. 282-291, March 1993.

[16] F. Signor and W. Snelgrove, "Switched-Capacitor Bandpass Delta-Sigma A/D Modulation at 10.7 MHz." IEEE Journal of Solid State Circuits, Vol. 30, No.3, pp. 184-192, March 1995.

[17] B. Song, "A Fourth-Order Bandpass Delta-Sigma Modulator with Reduced Number of Op Amps." IEEE Journal of Solid State Circuits, Vol. 30. No.12, pp. 1309-1315, December 1995.

[18] O. Norman, "A Bandpass Delta-Sigma Modulator for Ultrasound Imaging at 160 MHz Clock Rate." IEEE Journal of Solid State Circuits, Vol. 31, No.12, pp. 2036-2041, December 1996.

[19] S. Jantzi, K. Martin, and A. Sedra, "Quadrature Bandpass Delta-Sigma Modulation for Digital Radio," IEEE Journal of Solid State Circuits, Vol. 32, No.12, pp. 1935-1950, December 1997.

[20] A. Ong and B. Wooley, "A Two-path Bandpass Sigma-Delta Modulator for Digital IF Extraction at 20 MHz," IEEE Journal of Solid State Circuits, Vol. 32, No.12, pp. 1920-1934, December 1997.

[21] S. Rabii and B. Wooley, "A 1.8-V Digital-Audio Sigma-Delta Modulator in 0.8-um CMOS," IEEE Journal of Solid State Circuits, Vol. 32, No.6, pp. 783-796, June 1997.

[22] R. Schreier and M. Snelgrove, "Bandpass Sigma-Delta Modulation," Electronic Letters, Vol. 25, pp. 1560-1561, Nov. 1989.